(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
3 January 2003 (03.01.2003)

PCT

(10) International Publication Number
WO 03/001770 A2

(51) International Patent Classification[7]: H04M

(21) International Application Number: PCT/IB02/04148

(22) International Filing Date: 21 June 2002 (21.06.2002)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/299,745 22 June 2001 (22.06.2001) US

(71) Applicant (*for all designated States except US*): **EM-BLAZE SYSTEMS, LTD.** [IL/IL]; 22 Zarhin Street, P.O. Box 2220, Industrial Zone, 43662 Ra'anana (IL).

(72) Inventors; and

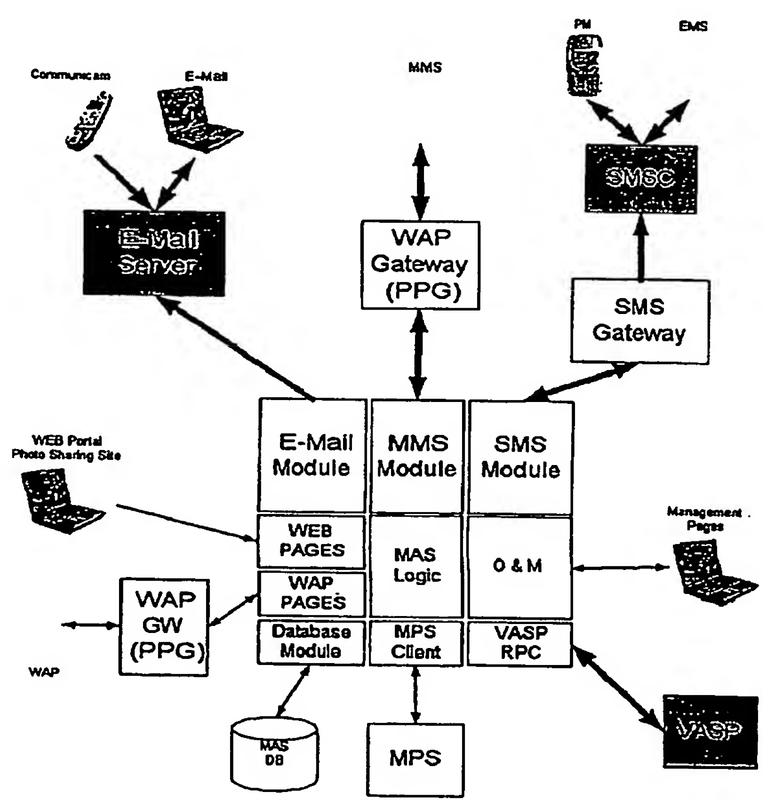
(75) Inventors/Applicants (*for US only*): **LEV, Tvsi, H.** [IL/IL]; Lessin Street 11, Apartment 6, 62997 Tel Aviv (IL). **AVNIMELECH, Ran** [IL/IL]; c/o UCNGO, Ltd., Aba Hilel Street 14, 52506 Ramat Gan (IL).

(81) Designated States (*national*): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, OM, PH, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZM, ZW.

(84) Designated States (*regional*): ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

[Continued on next page]

(54) Title: MMS SYSTEM AND METHOD WITH PROTOCOL CONVERSION SUITABLE FOR MOBILE/PORTABLE HAND-SET DISPLAY



(57) Abstract: An MMS communication system for displaying images on a display terminal of a mobile or portable communication device, the system comprising: an input adapted to receive pre-source information;a transmitter adapted to transmit the pre-source information;a server adapted to receive the transmitted pre-source information and further adapted to convert the pre-source information to source information suitable for display on the display terminal; and a source transmitter adapted to transmit the source information to the display terminal.

WO 03/001770 A2

Published:
— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

MMS SYSTEM AND METHOD WITH PROTOCOL CONVERSION SUITABLE FOR MOBILE/PORTABLE HANDSET DISPLAY

~~I.~~ DESCRIPTION

~~I.A.~~ *Related Applications*

[1]This Application claims priority from co-pending U.S. Provisional Application Serial No. 60/299,745 filed June 22, 2001, which is incorporated in its entirety by reference.

~~I.B.~~ *Field*

[2]This disclosure teaches techniques related to an MMS (Multimedia Messaging System), including images, graphics, numerics, and text, suitable for display on the display of a mobile or portable communication handset terminal.

~~I.C.~~ *Background*

~~1.~~ <u>Glossary of technical terms</u>

[3]To understand the disclosure better, the following definitions for technical terms used in this disclosure is provided:

[4]1. EMS: Extended Messaging System, a source protocol used for EMS handsets, designed to encode multimedia messages, including images, graphics, numerics, animations, audio and formatted text.

[5]2. MMS: A pre-source (multimedia+formatting information) protocol used to encode types of messages, including images, graphics, numerics, and text, and transcoded for the display/phone speaker on various display terminals. Used in most Nokia handsets.

[6]3. PM: Picture Messaging protocol, a graphic format (source protocol) used to display B/W images in Nokia handsets supporting the NSM per-source format.

[7]4. Pre-source information: In this application, information, which may be a full multimedia message or some part thereof, which appears in a non-source format and is not coded in a source protocol. Pre-source information refers to "packaged" multimedia content in a "raw" format such as:

[8]a. A set of TCP/IP packets composing a MIME multipart message (could be an email message or an MMS MM1 message), where some parts of the message are media objects which need to be converted/transcoded, and some other parts (e.g. SMIL attachment) are presentation layer information relating to how the information has to be arranged and displayed. Figa. 24 and 25 illustrate this concept.

[9]b. A block of SMS messages that together compose an EMS or a Nokia Smart Messaging (NSM) message and contain multiple media objects – pictures, ringtones, etc. These SMS messages are further encapsulated into the SMSC protocol which can be SMPP,UCP,CIMD etc.

[10] 5. Smart Messaging: A source protocol being developed by Nokia for Nodia handsets. This refers to everything defined in the NSM pre-source protocol and adds functionality for calendar events (vCalendar), electronic business cards (vCard) etc.

[11] 6. Source information: In this application, information, which may be a full multimedia message or some part thereof, which appears in a source format and which is coded in a source protocol. Typical source protocols are WBMP, EMS, and PM, but new protocols are being developed on an ongoing basis. Source protocols enable the display of messages on terminals with limit memory, processing, and display capabilities, such as those of mobile and portable radio communication handsets (i.e., cellular telephones, land mobile radios, Instant Messaging terminals, radio enabled PDAs, and the like). On the other hand, source information constitutes media objects in some media format, e.g. a JPEG picture, an MP3 audio file, an AVI video etc.

[12] 7. Transcoding: To perform protocol conversion, either from one source protocol to another, or from a pre-source protocol into a source protocol.

[13] 8. WAP: "Wireless Application Protocol", one protocol used for what have been called 2.5G cellular systems. In the cellular world, 1G was the original set of analog cellular systems. 1G has been mainly displaced by 2G systems, which are low-speed digital systems, which a typical raw data rate of 9.5kbps. Operators are currently deploying what are known as 2.5G systems, which are higher speed digital systems, expected to operate up 384kbps. 2.5G systems are expected to be replaced by 3G systems, which are higher speed digital systems promising speeds up to 2Mbps. WAP is one of the chief manifestations of 2.5G systems. WAP is a pre-source protocol.

[14] 9. WBMP: The display protocol for handsets in the WAP system.

~~2.~~ Introduction

[15] The process of transcoding is not a new idea. Indeed, in forms the basis of communication systems. Even the conversion from analog to digital, or vice verse, is a form of transcoding. With the proliferation of higher speed digital cellular systems, the challenge and the problem of transcoding have become much greater. There is, as yet, no standard display protocol for higher speed communication terminals. Therefore, transcoding from one display protocol to another is required to insure that the receiving terminal will be able to display the transmitted. There is, however, no method or system to do this in such a way that the integrity and quality of the transmitted message will be maintained in the display terminal. Further, there is no method or system for transcoding non-source information into a source protocol suitable for display on a communication terminal, while maintaining the integrity and quality of the information which originally appeared in the non-source format. There are transcoding systems and methods, to be sure, but they are primitive, and lose much of the quality of the source or pre-source information, even to the point where in some cases the displayed information in the display terminal is not recognizable. What is required is algorithms, a method, and a system, that will allow identification of the specific display characteristics of the target display terminal, and will also allow the source or pre-source information to be displayed on the target display terminal

with the maximum amount of integrity and quality in comparison to the pre-coded information.

~~II.~~ SUMMARY

[16] The disclosed teachings provide for:

[17] 1. Conversion of source information coded in a source format into a protocol suitable for transmission to and display on the terminal. A variety of new processing techniques are disclosed. Source information is typically coded in protocols such as WBMP (the protocol for Wireless Application Protocol, or "WAP", systems), EMS, and PM. This information must be transcoded for display on different terminals, also using source protocols, but where the protocols and variations of the protocols are typically different between the input source and the display terminal.

[18] 2. Conversion of pre-source information, that is, information which is coded but not in a source protocol, into a source protocol. For example, an ordinary digital picture will be transcoded into the source protocol WBMP. It will be appreciated that information in source protocol will then be transcoded again into the target source protocol, as explained in introductory point 1 immediately above.

[19] An MMS communication system for displaying images on a display terminal of a mobile or portable communication device, the system comprising: an input adapted to receive pre-source information;a transmitter adapted to transmit the pre-source information;a server adapted to receive the transmitted pre-source information and further adapted to convert the pre-source information to source information suitable for display on the display terminal; and a source transmitter adapted to transmit the source information to the display terminal.

~~III.~~ BRIEF DESCRIPTION OF THE DRAWINGS

[20] The above advantages of the disclosed teachings will become more apparent by describing in detail preferred embodiments thereof with reference to the attached drawings in which:

[21] FIGS. 1-37 show various features of the disclosed teachings as described in the rest of this document.

~~IV.~~ DETAILED DESCRIPTION

~~IV.A.~~ *Overall Architecture.*

[22] An implementation of the disclosed teachings is shown in FIG.5. The structure includes the input devices 5.11-5.13 on the left, the server 5.2 represented by the block in the middle, and the display devices 5.31-5.34 shown on the right. Information may be in source format, as is the case for the cellular telephone 5.12 (picture of two people) and the digital camera attached to a cellular telephone 5.13 (picture of the automobile). Or information may be in a pre-source format, such as the cartoon of the man 5.11. At the server, the source information or pre-source information is processed by a variety of components, adapted to

implement a variety of algorithms or techniques, which form an integral part of the disclosed teachings.

[23] Some of the components mentioned above perform at least the following tasks:

[24] 1. Format transcoding (from the pre-source information into source information, or from source information into other source information suitable for display on the display terminal);

[25] 2. Image adaptation, to adapt the image to the particular display screen on the display terminal. (This is discussed in further detail in section IVG);

[26] 3. Optimal compression for handset. The display terminal cannot display all of the bits of the original information. The information must be compressed, both for transmission and for display.

[27] 4. Photo enhancement: Specific sections of a photograph may be cutout and enhanced. (This is discussed in further detail in section IVG);

[28] 5. Content based processing, by which different aspects of a multimedia message are identified and processed differently. (This is discussed in further detail in section IVG);

[29] 6. Recognition from images: This is allied to the photo enhancement algorithm. Different portions of an image are recognized and "cutout" for enhancement. (This is discussed in further detail in section IVG);

[30] 7. Interfaces to 3rd Party Applications: Third party applications may be processed separately and sent to the display terminals, or may be added to the original information. In addition, if there are software packages with additional algorithms for additional processing of the source information, these may abe accessed and applied to the original information for eventual display on the display terminals. An example of an interface to a third party application is an XML-based interface over TCP/IP. Another example of such an interface would be and API in C++ or Java. By third party application we are referring to both external SW modules, and to VAS – value added services – e.g. a news service, a gaming platform for cellular phones (for example www.wirelessgames.com, www.cash-u.com), a photo album service. These applications wish to send MMS content and also in certain cases to perform special processing on such content prior to sending it. For a review of the special functionality please refer to section IV. H.

[31] Another implementation of the disclosed teachings is shown in FIG.6. Examples of input sources, called "Content Sources", 6.11-6.14, appear in the column at the right. However, the input sources are much broader than these pictures. At the bottom of the figure are various information devices 6.21-6.24 which can serve as both sources of information to the server 6.3, and also receivers of information processed by the server. These information sources can be WAP or i-Mode Phones, called "MMS Box" in the slide. (i-Mode phones are those that operate on i-Mode 2.5G cellular systems currently functioning in Japan. i-Mode phones will also operate on 3G systems expected to be introduced in Japan in late 2001 and in 2002.) Picture messaging phones, operating with the PM protocol,

are portrayed in the "Picture Box". Similarly, EMS phones, operating with the EMS protocol, are portrayed in the "EMS Box". Finally, Email enabled phones are portrayed in the "E-mail Box". In this implementation, information may be coded out of or coded into, any of the protocols show, including WAP phones (WBMP protocol), I-Mode Phones (the Japanese version of WAP), Picture Messeging Phones (PM protocol), EMS Compliant Phones (EMS protocol), and E-mail Capable Phones (POP3, SMTP, and IMAP4 protocols)

[32] The server will receive, transcode, and optimize for display on a specific communication terminal, at least any of the following protocols: IP, SMPP, TCP/IP, POP3/SMTP/IMAP4/XML. This is illustrated in FIG.7

[33] Using the transcoding several tasks are accomplished, for example:

[34] 1. The conversion from and into any of the various formats WBMP, PM, and EMS.

[35] 2. The conversion of formats on the fly when the user logs on to his or her MMS box (so that even if the user were to switch terminal devices, he or she would get the correct content, properly formatted).

[36] 3. The conversion taking into account the exact parameters (such as, for example, screen size, pixel dimensions, etc.) of the particular terminal and terminal display.

[37] 4. Transcoding of source information into other source information suitable for display on a target terminal, maintaining integrity and quality of the original message. examples would be:

[38] a. A JPEG image is converted to a GIF image so that a phone with a WAP browser than can display GIF images will be able to view it.

[39] b. A Nokia ringtone is converted to an EMS iMelody ringtone so that a non-Nokia phone can play it.

[40] c. A video in MPEG1 format is converted to MPEG-4 Simple Visual profile so that an MMS compliant phone can display it, or to an animated GIF sequence so that a non-MMS compliant (legacy) phone can view it.

[41] d. Formatted text in the EMS format can be converted to an image or to HTML+text to preserve the formatting (underline, bold, letter size etc.).

[42] It should be noted that both the EMS and NSM formats include not only images but also ringtones, animations and formatted text. This fact is well documented in the EMS/NMS standards documents for the last few years.

[43] 5. Transcoding of non-source information into other source information, for later transcoding into source information for display on a specific target terminal, where all transcoding maintains the integrity and quality of the original message.

[44] 6. Recognizing the specific display characteristics of a specific target display terminal, to enable the display of high quality messages, whatever the characteristics of the terminal.

~~IV.B.~~ An Example Implementation: Overview

[45] FIG.8 shows another example implementation of the overall system. This implementation is called MMR. The MMR system allows users to send and receive messages containing text and images at least in the following formats / protocols: WAP; PM; EMS; MMS; E-MAIL; WEB; SMS.

[46] While the term message and information has been used in discussing the implementations in detail, it should be clear to a skilled artisan as to which message/information constitutes pre-source information and which constitutes source information according to the definitions for the same provided in the background.

~~1.~~ User WAP Pages

[47] The MMR provides a WAP based messaging application, allowing users to login to their personal messaging page. From this page users can view and send messages in variety of formats. The MMR sends the WAP recipient an SMS notification with a link to the newly received message. Alternatively, the MMR can send a WAP push message with the same link. MMS recipients receive a notification to the new MMS. PM and EMS recipients receive the message directly. Email recipients receive an e-mail with an image attachment to their regular e-mail address.

[48] It should be noted that WAP pages can contain multimedia for immediate integrated display (e.g. a WBMP image), but can also contain downloadable multimedia such as higher resolution images, ringtones, animations etc. , as part of e.g. the M-Services standard for downloadable media.

~~2.~~ User Web Pages

[49] The MMR provides a WEB based portal for three major function.

[50] 1. Users can register themselves to the system, by submitting personal information as well as information about the model of their mobile device.

[51] 2. A photo album application is provided for personal storage and sharing of images and audio files. Users can login to their personal accounts, view and send messages to mobile device, in the same manner as described above.

[52] 3. Users can also login to an HTML based messaging page which allows them to view all their received messages. Regardless of the format in which these messages were originally sent, these messages will be transcoded to be viewed on a normal web browser. This may allow users to view messages in a much higher quality than that seen on their mobile devices.

~~3.~~ MMS Module

[53] The MMR can send and receive MMS messages to and from mobile devices. Incoming MMS messages are parsed and transcoded for optimal display on the recipient's device. Recipients of outgoing MMS messages receive a notification, allowing them to download the message from the MMS proxy. Other recipients receive the MMS message after it has been transcoded to WAP, PM , EMS , SMS or E-Mail.

~~4.~~ Email Module

[54] The MMR allows users to send E-mail messages with image attachments to mobile devices. It also allows mobile users to send E-Mails with image attachments to regular e-mail addresses. Incoming E-mails are parsed. The e-mail subject is sent as the message text, while each of the image attachments in the original e-mail are transcoded for the mobile device. Depending on the amount of attachments, the recipient may receive several messages, and a format most suitable to his mobile device. Outgoing e-mail messages use SMTP to send the message text along with an image attachment to the selected recipient (any e-mail address).

[55] It should be noted that the e-mail interface is also utilized for sending images from an Ericsson Communicam to other mobile devices. Communicam images are posted from the camera to a dedicated server, which converts these images to an e-mail with image attachments. Proper configuration of the e-mail recipient address allows the user to send these images to other mobile devices. Communicam is a specific commercial line of cameras that can be attached to phones. It is referred here to a general camera attached to a phone.

~~5.~~ SMS Module

[56] The MMR allows users to send messages in several SMS based formats. Picture Message for Nokia phones, and EMS messages for Ericsson phones are supported.

[57] Incoming messages are transcoded into PM and EMS, dividing the original message into up to 6 SMS messages. The recipient's phone receives these SMS messages, and concatenates them. When all SMS messages have arrived, an application on the mobile device displays the message content.

[58] It should be noted that MMR can also receive PM and EMS messages originating from mobile devices, and transfer these messages in different formats to other devices. This feature requires a special agreement between the SMS service provider and the MMR operator, in order to forward all concatenated SMS messages through the MMR.

[59] The conversion a full message (pre source with source objects) is a conversion where certain constraints and relations between the media objects requires more processing and application of "business rules": for example, if an EMS message which is 6 SMS long is sent to a Nokia phone (NSM messages are up to 3 SMS long) some or all of the following operations may take place:

[60] a. The images get resized to reduce their size in bytes.

[61] b. The Audio files get truncated to reduce their size in bytes.

[62] c. The text formatting may be removed to reduce total message size.

~~6.~~ MMR Logic

[63] The MMR-Logic controls the behavior of the MMR. Using the MMR database, and set of configurable rules, the MMR Logic selects the most suitable message format for each recipient. It then determines the correct data flow path for each of the possible message transactions. The MMR-Logic is also responsible for the on-the-fly gathering of information about users and their mobile devices. This is performed by e.g. registering the WEB/WAP user-agent of the phone when it sends requests, or by identifying the type of the message sent from a phone (e.g. an EMS message) which indicates that this phone can send/receive message in this format.

~~7.~~ MMR Database

[64] The MMR database stores information about the system users, such as name, phone number, phone model etc. Message contents, i.e. images, audio and text, are also stored in the database. The MMR database also contains information required by the O&M block.

~~8.~~ O&M

[65] The MMR O&M functions provide the MMR system administrator with an array of tools to monitor and control the behavior of the MMR. A Web based interface, provides the administrator access to web pages, arranged in groups according to the functional blocks in the MMR.

[66] The O&M also provides the administrator with a messaging page, which allows him to send messages in all formats to mobile devices.

~~9.~~ MPS Client

[67] The MPS client translates the required transcoding action, as determined by the MMR-Logic block, into an XML request. This request is then posted to the MPS server rack. The MPS client then parses the response, and extracts the transcoded image. Further details are found in section IV.I

~~10.~~ MPS (Media Processing Server)

[68] The media-processing server handles the message transcoding from one format to the other. Other image processing functions such as face detection can also be called via the XML interface.

~~11.~~ VASP RPC Module

[69] The MMR provides an external interface to Value Added Service Providers (VASP), allowing remote invocation of MMR functions via an XML RPC interface.

[70] Tthe XML RPC infrastructure can be easily expanded to include additional remote procedure calls. Details are found in sections IV H and the last section titled MPS control interface document.

~~12.~~ Internal WAP GW

[71] The MMR hosts an internal WAP gateway. This gateway is required to support functionality not yet supported on commercially deployed gateways. The internal WAP gateway allows the MMR to send/receive MMS messages, as well as use WAP push for message notifications. The inclusion of an internal WAP gateway is optional, not a must. An external MMS compliant WAP gateway supporting segmentation and re-assembly (SAR) and MMS tags/mime types can be used.

~~13.~~ Internal SMS GW

[72] The internal SMS GW is used due to special functionality required for receiving EMS and PM messages. The SMS gateway is an interface layer/mediator for receiving and sending the SMS messages from/to an SMS center (SMSC) via the prevailing protocols such as UCP,SMMP,CIMD etc.

~~IV.C.~~ Details of Selected Functional Blocks

~~1.~~ User Web Pages – The MMR Web Site

[73] The public MMR main web portal contains links to at least the followingr functions:

[74] Link to the user's personal web based "Messaging Application"

[75] Link to the "Photo Album"

[76] Link to the "User Registration Page"

~~a)~~ Web-based Messaging Application

[77] The web based messaging application provides similar functional capabilities to the wap based application. User's may enter their personal messaging page, by using the same user name and password as used on their mobile phones. Once inside, user's can view and send messages in a variety of formats.

[78] From the web-based application, users can also send messages with original content.

~~b)~~ Web-Based Photo Album

[79] The MMR provides a web based photo album application, allowing the user to upload and manage their own folders containing images and audio files. These files can then be shared with friends, or sent to mobile devices in a variety of formats.

[80] The MMR can automatically select the message format most suitable for the recipient, or may receive a request from the sender to send the message in a specific format.

~~c)~~ Web-BasedUser Registration Page

[81] The user registration page allows new users to register themselves to the service. It also allows registered user's to update their registration information. Registration information requires the user to submit some personal details, as is accustomed in web based email services. In addition to this information, the user can be asked to submit information regarding the model of his mobile device.

~~2.~~ The E-Mail Module

[82] The MMR allows users to send E-mail messages with image attachments to mobile devices. It also allows mobile users to send E-Mails with image attachments to regular e-mail addresses. Incoming E-mails are parsed. The e-mail subject is sent as the message text, while each of the image attachments in the original e-mail are transcoded for the mobile device. Depending on the amount of attachments, the recipient may receive several messages, and a format most suitable to his mobile device. Outgoing e-mail messages use SMTP to send the message text along with an image attachment to the selected recipient (any e-mail address).

~~a)~~ Email Server Account Set-Up

[83] The e-mail server needs to be configured to receive all mails addressed to a selected domain , e.g. pics.ucngo.com . All incoming e-mail messages in this format are accepted by the e-mail server. Furthermore, the server is configured to create an event for each incoming message. This event triggers the MMR to handle the new message an described in the sections below.

~~b)~~ Email to Mobile Device

[84] 1. Receive messages sent to : user-phone-number@pics.ucngo.com e.g. 97254985026@pics.ucngo.com

[85] 2. The e-mail is handled as follows:

[86] i) The email subject is extracted, and used as the message text. (The message text is limited to 120 characters (configurable)).

[87] ii)The phone is extracted from the email address and used as the recipient's number.

[88] iii) The Image attachments are used as the message images. Each attached image generates a new mobile message, with the same message text. The maximum allowed message size is 150Kbyte (configurable)

[89] 3. The message is stored in the message table of the MMR database.

[90] 4. The MMR-Logic selects the optimal message format for the recipient.

[91] 5. A message in the selected format is sent to the recipient.

~~c)~~ Outgoing Error Messages

[92] Incoming e-mail messages that can not be handled according to the logic above, will generate an error message. This error message will be sent to the e-mail originator, to notify him that his message could not be handled. For several expected cases, the exact error will be given, to explain why the message could not be handled:

[93] i) Recipient number is not a valid number, or is unknown to the system.

[94] ii) Message does not contain a valid image attachment.

[95] iii) Message attachments are larger than the allowed quota.

~~d)~~ Mobile Device to E-mail

[96] Mobile devices may send e-mail messages via the WAP messaging portal.

[97] During the process of sending a message, the WAP user is provided with a "send as" link, allowing him to select from a list of optional formats.

[98] By selecting "send as e-mail" the user prompts the following chain of events:

[99] i) A new e-mail message is composed.

[100] ii) The image original is taken from the message database, and sent as an e-mail attachment.

[101] iii) The message text, as edited by the user is sent as the e-mail body.

[102] iv) The e-mail subject is a generic message in the form of: "You have received a new mobile e-mail from <sender number>"

[103] v) The recipients e-mail address can be entered in one of two ways:

[104] 1. If the e-mail recipient is a registered user, the sender may type in the recipient's phone number, and the MMR will lookup the recipient's e-mail address from the database.

[105] 2. If the recipient is not registered, or if his e-mail address is not known, the sender will be directed to a wap page from which he can edit the required e-mail address.

~~3.~~ <u>SMS based Messaging</u>

[106] The SMS-based module is in charge of generating at least the following message format: EMS, PM, WAP link notification (SMS or WAP-push), WAP-push MMS notification, Text SMS for devices that do not support images.

[107] Furthermore, the SMS module includes an SMS link layer, capable of receiving EMS and PM messages from mobile devices. The link layer can then concatenate the fragmented SMS messages that make up an EMS or PM and extract the message image and text. These messages can then be transcoded into any of the supported formats.

[108] The MAS core is a group of java servlets, that handle image transcoding management / message transfer / database functions / billing and O&M functions.

[109] These servlets have external interfaces to an Email server , SMS center and WAP / WEB gateway allowing the MAS to interconnect between devices using these protocols. Refer to the : SMS / EMS / PM Module for a more detailed block diagram of the SMS / EMS / PM modules. Refer to the : MAS E-Mail Module for a more detailed block diagram of the POP3 / SMTP modules.

[110] The PM/EMS/SMS receive will be handled by a dedicated servlet, it will interface all incoming SMS's handled by the SMS GW. It will encode the incoming SMS's using the following top level logic:

[111] 1) Detect type of message single, concatenated

[112] 2) For Single Message:

[113] a) Detect Type of message Text, PM, EMS

[114] b) Extract Image or Text from message

[115] c) Post data to with text & phone number to sms handler servlet

[116]

[117] 3) For Concatenated Messsage Store in local db with message ID

[118] 4) When Last message received – (from analyzing XX,YY,NN : XX – msg id, YY - total number of msgs, NN – msg sequence in the UDH) and tracckin sequence of received messages, do:

[119] a) Concatenate message data

[120] b) Detect Type of message Text, PM, EMS

[121] c) Extract Image or Text from message

[122] d) Post data to with text & phone number to sms handler servlet

~~4.~~ <u>MMR-MM1 System Logic</u>

[123] The MMR Logic module determines the data flow path and transcoding type used on messages that go through the system. Sub-section 4(a)defines the chain of events that take place, for each of the possible combinations of input and output formats. However, there are cases where the recipients phone capabilities are either not fully known, or the recipient's phone may be able to accept messages in more than one format.

[124] Subsection 4(b) deals with selecting the correct message type for the recipient.This sub-section deals with scenarios where either the sender or recipient's information is either not known to the system, or it conflicts with previous information stored in the MMR about the user.

~~a)~~ <u>Transcoding Matrix</u>

[125] The MMR enables messages to be sent from one device to the other, automatically transcoding the message content from the source device format to the target device format.

[126] The supported formats are : WAP / WEB / E-Mail / PM / EMS / MMS / SMS.

[127] The following sub-sections describe some of the various transcoding actions taken for each combination of source and destination formats.

~~(1)~~ WAP to WAP

[128] Image source data is already in the database.
[129] Send notification to the recipient with a link to the new image.
[130] Transcode the image when the recipient enters the message page according to the UA used.

~~(2)~~ WEB to WAP

[131] Upload new image via HTTP to the database.
[132] Send notification to the recipient with a link to the new image.
[133] Transcode the image when the recipient enters the message page according to the UA used.

~~(3)~~ WEB to Preview

[134] Upload new image via HTTP to the database.[135] Transcode the image to the expected recipient phone type.
[136] Transcode the image from the format above to the PC monitor format.
[137] Display resulting image wrapped in some html.
[138] Delete the image original from the database.

~~(4)~~ E-Mail to WAP

[139] Extract images and recipient numbers.
[140] For each recipient send all the images.
[141] Send notification to the recipient with a link to the new image.
[142] Transcode the image when the recipient enters the message page according to the UA used.

~~(5)~~ WAP to E-Mail

[143] Take image original from the database.
[144] Transcode to GIF or JPEG to a size no larger than 25Kbyte (Configurable)
[145] Send to recipient's e-mail address.

~~(6)~~ WAP to PM /EMS / MMS

[146] Take image original from the database.
[147] Transcode to PM / EMS / MMS according to recipient phone type.
[148] Send message.

~~(7)~~ WEB to PM /EMS / MMS

[149] Upload image from web or photo-sharing site.
[150] Transcode to PM / EMS / MMS according to recipient phone type.
[151] Send message.

~~(8)~~ E-Mail to PM / EMS / MMS

[152] Extract images and recipient numbers.
[153] For each recipient send all the images.
[154] Transcode the image to PM / EMS / MMS according to recipient's phone type.
[155] Send message.

~~(9)~~ WAP / WEB / E-Mail to SMS

[156] This mode will be used when the recipient's phone cannot display images.
[157] The message text will be sent as an SMS to the recipient.

~~(10)~~ EMS / PM to WAP *

[158] Receive and store EMS/PM as fragmented SMS messages.
[159] Link fragments to a complete EMS/PM
[160] Send notification to the recipient with a link to the new image.
[161] Transcode the image when the recipient enters the message page according to the UA used.

~~(11)~~ EMS / PM to EMS / PM *

[162] Receive and store EMS/PM as fragmented SMS messages.
[163] Link fragments to a complete EMS/PM
[164] Create and send an EMS or PM according to the recipients known device capabilities.

~~(12)~~ EMS / PM to MMS *

[165] Receive and store EMS/PM as fragmented SMS messages.
[166] Link fragments to a complete EMS/PM
[167] Transcode EMS/PM into MMS format.
[168] Initiate an MMS transaction with the WAP Gateway.

~~(13)~~ All to SMS

[169] Strip images from the original content.
[170] Send the first 160 characters of the text message as an SMS message.

~~(11)~~ MMS to all formats

[171] Receive the MMS message. Extract the content files and the SMIL.
[172] Insert the all image and audio files into the database.
[173] Enter the SMIL file into the database. Transcode the SMIL informatin into HTML/WML/EMS formatting information if the targets are WEB,WAP,EMS respectively. If the target is email or an MMS phone that does not support SMIL, the media objects (MIME objects) may be reordered based on the information in the SMIL description to ensure proper viewing order between the various media objects.
[174] Send the message to the selected output format, as if it came from the WEB. * - Requires redirection of concatenated SMS messages by the SMSC through the MMR. The MMR then handles EMS/PM messages according to the above logic.

~~b)~~ Sender and Recipient Logic

[175] In order to properly perform a message transaction, full knowledge is required about the sender and the recipient. This is required to allow for optimal transcoding of the input message to the correct output format. However, there are cases where either the sender or the recipient (or both), are "not completely" known to the MMR. The data stored in the MMR database may be incomplete or inaccurate. For example, a user might be registered to the service, without the MMR knowing which device is being used. This information may also change when the user moves his SIM card from one device to another. In other cases, the user might not be registered in the database at all. The purpose of this module is to perform the correct logic decisions, to make the most out of the data that is known to the system. Furthermore, the sender and recipient logic are used to gather information about the system users in an un-formal way, by correlating information such as phone numbers, device user agent, and incoming message formats. This information is added to the information submitted by the user, during the registration to the service (which is not mandatory, but recommended). Further details are included in Section J.

~~c)~~ On-The-Fly Data Collection

~~(1)~~ PM / EMS / MMS Capability. (PM and EMS input Disabled)

[176] When an unregistered or a partially known user sends a PM , EMS , or MMS message to an MMR recipient, the MMR can register the sender on the fly. The purpose of this action is to update the database, and add users on the fly. If the user was already registered, the MMR checks that the user's capability to send messages in this format is already known.

~~(2)~~ Updating the User's WAP Profile

[177] When a registered user sends a WAP message to an unregistered recipient, this recipient receives an SMS notification. When the recipient enters the message page, his phone's User-Agent becomes known to the system. At this point the MMR can add the recipient as a new user, and assign the correct device to him. This function is also useful for registered users who are now using a new phone model. The database can be updated with the new user agent. Accordingly, other capability flags, such a PM and EMS might now change. The exact same description also applies for phones using HTTP (standard WEB) browsers – there too the browser specifies a User Agent.

~~(3)~~ Synchronizing the User data and device data

[178] At any given moment, the MMR database might hold information about the user and his mobile device. Since some of the message formats may operate by using the user's capability flags alone, some users may not have a registered device type for extended periods. When user's register themselves through a dedicated registration process, or when users enter a WAP session their device becomes known. At this point it is important to verify that there is no discrepancy between the user's capability flags, and the devices' capability flags. The synchronizing process forces the devices' capabilities on the user.

~~d)~~ Selected Message Type Logic

[179] This section explains the logic implemented in the MMR Logic Module, to select the correct message type for the recipient. The logic is divided into a case where the recipient is registered (at least with partial data), or when the recipient is unknown to the MMR.

[180] For each transaction, this logic should bring the system to the following state:
[181] There is a valid sender data structure.
[182] There is a valid sender device data structure.
[183] There is a valid recipient data structure.
[184] There is a valid recipient device data structure.
[185] The format of the incoming message is well defined.
[186] The format of the required output message is well defined.
[187] The flow of events required to carry out this transaction is well defined as discussed above in 4(a).

~~(1)~~ Selected Message Type for a registered recipient

[188] The selected message type for the recipient for any transaction is effected by the following parameters:
[189] A direct request by the sender for a specific message type.
[190] The user's device capability flags.
[191] The user's capability flags. (if the device is not known)
[192] The user's preferred message type.
[193] The device's preferred message type (if the user did not specify one)
[194] The original message format.
[195]
[196] The capability flags show if the user can accept message in the following formats:
[197] MMS / LEMS / SEMS / PM / WAP Flag.
[198]
[199] Due to the information required before a decision can be made, the recipient's data must be known. The receiver data may either be known because it was stored in the database, or because it was temporarily created for this transaction, as explained in section 4(b). In any case, at this point there can no longer be a discrepancy between the user's capability flags and his device's capability flags.
[200] Given that the recipient is known, the selected message type will be chosen according to the following logic.
[201] If the sender requested a specific format, that format is selected. (Forcing the format by the sender may result in the message not being sent. This is not the normal mode of operation. In the normal mode, the sender selects "automatic" and the MMR decides the best format automatically.)
[202] If the sender mode is automatic, the user's "preferred message type" is compared to the devices / user's capability flag. If it is a legal selection, the message is sent to the user in his preferred format.
[203] If the user has no specific preference, and user's device data is the next dominant information according to the following logic:
[204] o If one of the optional formats of the device allows the message to be sent without being transcoded, that format is selected.
[205] o If the message must be transcoded, and the device has a "preferred format", that format is chosen.
[206] o If the device data doesn't specify a "preferred format", the best of the format options is selected according to the following order: MMS, WAP, EMS, PM.
[207] o If the user's device is not known, the user's data is the next dominant information according to the following logic:
[208] o If one of the optional formats acceptable by the user allows the message to be sent without being transcoded, that format is selected.
[209] o If the message must be transcoded, the best of the format options is selected according to the following order: MMS, WAP, EMS, PM.

~~(2)~~ Selected Message Type for an unregistered recipient

[210] If the sender requested a specific format, that format is selected. (Forcing the format by the sender may result in the message not being sent.)
[211] If the sender did not request a specific format, the message will be sent according to a default table. The default table will be able to select a default output format for each input format. This table will be configurable with a simple editor. Table 1 is an example of the default output format table.
[212] A temporary recipient data structure is created with coherent information to allow the selected transaction to take place.

[213]

[214]	Input Format		WAP	WEB	E-Mail	PM	EMS	MMS
[215]	Default Output Format	WAP	WAP	WAP	PM	EMS	MMS	

[216] Table 1 : Default output format table.

~~5.~~ Transmitting pre-source information to server

[217] The pre-source information is transmitted to the server. The following description along with the figures 9-11 referred to herein provides, to a skilled artisan, further explanation on the transmission of pre-source information.

~~a)~~ User Control Web Page

[218] This is a site that will allow users to change the attributes the system holds about their phone. Authentication will be done as follows:

[219] User enters a page.

[220] Enters his phone number.

[221] System sends an SMS with a 4 digit code to the user while he is surfing.

[222] User will be able to change his phone type, EMS capabilities etc.

[223] User pressed submit.

[224] System prompts the user to get the secret code from the SMS inbox.

[225] User will enter the code and his registration details will be changed.

~~b)~~ JSP Pages

[226] The following sections are pseudo-code descriptions of the JSP pages.

~~(1)~~ login.jsp

[227] Enter your Phone number

[228] Enter your Password

[229] Struct userstruct = GetUserStructByPhoneNumber(number)

[230] Bool isvalid = AuthenticateUser(number, userstruct.password)

[231] If (isvalid) Goto Main.jsp (2.3) else if (Try again ?) goto login.jsp

[232] If ("Password" || "New user") Goto Forgotornew.jsp

~~(2)~~ forgotornew.jsp

[233] Enter phone number

[234] Bool newuserok = adduser(number, user-agent)

[235] If (newuserok) Bool passwrdsetok = SetNewPassword(number)

[236] If (passwrdsetok) userstruct = GetUserStructByPhoneNumber(number)

[237] Bool smssent = SendSMS(userstruct.password , number)

~~(3)~~ main.jsp

[238] Bool gotmsgvect = GetUserMessageIdVector(const number, msgidvect)

[239] If (gotmsgvect) numofmessages = msgidvect.size

[240] Set number in the brackets (i.e. Messages)

[241] Bool gotarchivevect = GetArchiveNameVect(archivevect)

[242] Int numofarchives = archivevect.size

[243] Set number in the brackets (i.e Archives)

[244] If "Messages" goto messages.jsp

[245] If "Archives" goto archives.jsp

~~(4)~~ messages.jsp

[246] for (I=0 ; I = numofmessages) CreateLinkToMsg(msgidvect[I])

[247] if "Message" pressed, goto msg.jsp (2.5)
[248] if "Delete All" pressed Bool deleteOK = DeleteAllMessages(msgidvect)
[249] if (deleteOK) goto MessagesDeleted.jsp, and then goto main.jsp

~~(5)~~ msg.jsp

[250] msgStruct = GetMessageStruct(msgidvect[I]) (including GetImageFileID)
[251] String msgtxt = msgStruct.txt
[252] Vector < char> image_buffer = msgStruct.GetImageBuffer( )
[253] DevStruct SourceDeviceStruct = GetUserDeviceByNumber(msgStruct.From)
[254] String UA =GetUserAgentStringFromSession( )
[255] DeviceStruct TargetDeviceStruct = GetUserDeviceByUserAgent(UA)
[256] XMLreq = GetXMLrequest = (SourceDevice, TargetDevice , image_buffer)
[257] XMLresponse = PostXMLrequest(XMLreq)
[258] Out_image_buffer = GetImageFromXML (XMLresp)
[259] Output message.wml
[260] If "Send" goto send.jsp
[261] If "Delete Message" Bool msgdeleteOK = DeleteMessage(msgStruct.msgID)
[262] If (msgdeleteOK) goto MessageDeleted.jsp, and then goto messages.jsp

~~(6)~~ send.jsp

[263] Edit text edit box.
[264] Edit number edit box.
[265] If "Send as" pressed goto #sendas card. Select message type.
[266] MsgStruct = CreateMessageStruct(From, To , type , text , imageFileID)
[267] If "Send" Bool sentOK = SendMessage(msgstruct)
[268] If sentOK goto sentok.wml and then goto messages.jsp
[269] If ((!sentOK) && (type=wapwap)) goto sendfailedjsp.jsp
[270] If ((!sentOK) && (type=wapemail)) goto noemailaddress.jsp
[271] If ((!sentOK) && (type=wapemspm)) goto notemsable.jsp

~~(7)~~ Noemailaddress.jsp

[272] if "Enter Address" goto Enteremail.jsp

~~(8)~~ Enteremail.jsp

[273] MsgStruct.emailaddress = emailaddress.
[274] sentOK = SendMessage(msgstruct)
[275] if "fix address" goto entermail.wml

~~(9)~~ archives.wml

[276] for (I=0 ; I = numofarchives) CreateLinkToArchive(archivevect [I])
[277] if "Archive" pressed, goto archive.jsp (2.10)

~~(10)~~ Archive.jsp

[278] Bool GotVector= GetArcMessageIDVector(ArchiveID , msgIDVect)
[279] for (I=0 ; I = numofmessages) CreateLinkToMsg(msgidvect[I])
[280] if "Message" pressed, goto msg.jsp
[281] if "Delete All" pressed Bool deleteOK = DeleteAllMessages(msgidvect)
[282] if (deleteOK) goto MessagesDeleted.jsp, and then goto main.jsp

~~e)~~ MAS-MPS client

[283] The MPS client block enables MAS servlets to use MPS transcoding services, as well as supplying an API for XML and Base64 functions. Listed below are the main functionalities of this block.

[284] Encoding and Decoding of images from binary to base64 ascii.

[285] Creating XML transcode request.

[286] Posting XML requests to an MPS server rack.

[287] Receiving and analyzing XML responses.

[288] Handling MPS errors.

~~IV.D.~~ Details of Media Processor

[289] The MEDIA Processor provides image processing and transcoding for purposes of image enhancement and terminal compatibility. Naı ve transcoding may result in unreadable content on the small screen of a mobile terminal. The Media Processor enhances the image to correct such faults when the content type is identified. the MPS also supports audio, ringtones, animation, video see for example AudioTranscode.

[290] Communication with the Media processor is implemented using XML interface. The Media Processor reports success or failure for an entire message as well as for each individual operation of the message. The media processor supports processing multiple images within a single message.

[291] At least the following media processing functions are available to render message images for display on user's device:

[292] Adaptation functions-- media format convert- from (Progressive JPEG, Baseline JPEG, JPEG 2000, GIF87, GIF 89A, WBMP, BMP, PNG, EMS, Nokia PM) to (Progressive JPEG, Baseline JPEG, JPEG 2000, GIF87, GIF 89A, WBMP, BMP, PNG, EMS, Nokia PM) including colour palette adaptation, all based on a client submitted device type parameter.

[293] Image content selections are provided to identify the type of image (e.g. -- Photograph, Face, Document (e.g. FAX), cartoon, Synthetic (e.g. chart), Panoramic (e.g. scenery).

[294] The MEDIA Processor includes a facility to smart compress images (VGA picture with smart JPEG compression takes maximum storage of approximately 50k).

[295] The Media Processor is capable of being shared by multiple clients.

~~1.~~ Enhancement functions

[296] The media processor provides the following media processing image enhancement functions:

[297] Brighten (dark), Darken (overexposed), Enhance, Colour balance, Remove Noise, threshold(local adaptive, standard), adjust levels, sharpen (radius, intensity, automatic), de-blur, smooth, histogram equalise, invert, flip(mirror), crop(arbitrary or parametric), Remove artefacts, resize(nearest neighbour, bi-cubic, bilinear, maximum/minimum neighbour, line preserving), salt and pepper removal, local illumination correction (arbitrary, emphasise edges), histogram equalisation,

histogram manipulation, Brightness, Contrast, Colour modification, Rotate (90, 180 or 270 degrees).

~~2.~~ Auto-Enhancement functions

[298] The media processor provides the following media processing auto-enhancement functions:

[299] Auto level, auto crop, auto colour balance, auto image content type detection, Image classification and Optimisation Processing. Add text as graphic, add background, image manipulation (warping), image framing, combine images, Add objects (hair, eyeglasses etc.), Include image in postcard / template, Camera calibration for common mobile camera types.

~~3.~~ Image Stitching

[300] The media processor provides the following image Stitching: stitch 360-degree panoramic and stitch fax. Full stitching 2 images / arbitrary-length series, Image pair matching, Image merging, given shift parameters, Image stitch/match given assumptions (e.g. horizontal only), stitch (Brightness, Contrast, Colour).

~~4.~~ Advanced Functions

[301] The media processor provides the following media processing advanced functions:

[302] Detect face; detect eyes, OCR Recognition, Bar code Recognition, picture object recognition, Image recognition (e.g. content type recognition to permit optimal transcoding).

~~5.~~ Watermarking

[303] Watermark detect and add functions shall be provided for WBMP and JPEG images. A watermark shall support a minimum of 19 decimal digits.

[304]

~~*IV.E.*~~ *Identifying display characteristics*

[305] The following code segment explains display characteristics identification.

[306] - <target-device>
[307] - <platform>
[308] <manufacturer>Ericsson</manufacturer>
[309] <model>R320</model>
[310] <ROM-revision>n/a</ROM-revision>
[311] <User-Agent>EricssonR320/R1A</User-Agent>
[312] </platform>
[313] - <network-connection>
[314] <nc-type>GSM/CSD</nc-type>
[315] <nc-speed>9600</nc-speed>
[316] </network-connection>
[317] - <target-display>
[318] <horizontal>88</horizontal>
[319] <horizontal-scroll>88</horizontal-scroll>
[320] <vertical>52</vertical>
[321] <vertical-scroll>110</vertical-scroll>
[322] <dpi>n/a</dpi>
[323] <pixel-ratio>1.24</pixel-ratio>

[324] - <colors>
[325] <type>B/W</type>
[326] <number>2</number>
[327] <bit-arrangement>n/a</bit-arrangement>
[328] <palette-LUT>n/a</palette-LUT>
[329] <gamma />
[330] <brightness />
[331] </colors>
[332] <emstype>none</emstype>
[333] <pdu>1300</pdu>
[334] <maxwpdksize />
[335] <maxpixels />
[336] </target-display>
[337] </target-device>
[338]

~~IV.F.~~ Additional Processing by Media Processor.

[339] The Media Processing Server (MPS) is designed to handle all media types, including formatted text, images, animations, audio and video, with an emphasis on advanced processing algorithms. In a nutshell, some of the following funcitonalities are provided:

[340] 1. Image Transcode – Optimally convert content for a target phone. Automatically performs resizing, color palette reduction, compression, rotation, watermark detection and more. The transcode operation is controlled by a rule based system with configurable parameters for bandwidth utilization, format usage, Quality of Service and content preferences. Performs different transcoding operations based on automatic detection of the content type.

[341] 2. Audio Transcode – Similar to transcode for audio files. Useful for converting audio found on the Internet to MMS phones. Also supports conversion of ringtones between the different formats existing today.

[342] 3. Video Transcode – similar to image transcode for video files. Also supports cross media conversion – video to animation, video to still image, video to sound track.

[343] 4. Image manipulation package:

[344] 4.1. Rotate – rotates an image by a specified amount with a selection of interpolation methods.

[345] 4.2. Resize – resizes an image with several interpolation methods including special modes for phone screen with a small number of colors/non-square pixels

[346] 4.3. Brighten – enhances the image brightness – useful for dark images and for adapting to phones with a nonlinear Gamma curve.

[347] 4.4. Darken - decreases the image brightness – useful for over-exposed images and for adapting to phones with a nonlinear Gamma curve.

[348] 4.5. Enhance – combines color and contrast enhancement of an image.

[349] 4.6. Color balance – performs color balancing of images taken by low quality cameras or in difficult lighting conditions.

[350] 4.7. DenoiseSpeckle – noise removal for low-light/noisy camera/data trasnsmission errors situations.

[351] 4.8. Threshold – binarization of images for B/W screens.

[352] 4.9. Adjust levels – parametric contrast adaptation/

[353] 4.10. Sharpen – fast parametric correction of blurry images.

[354] 4.11. Deblur – special sharpening for camera images taken in low light conditions.

[355] 4.12. Smooth – smooths a noisy image.

[356] 4.13. Histogram equalize – automatic contrast range enhancement.

[357] 4.14. Invert – performs a color/grayscale inversion. Useful for certain synthetic images on low contrast phone screens.

[358] 4.15. Flip – fast mirroring operation.

[359] 4.16. Crop – cut a part of the image.

[360] 4.17. ArtifactRemove – JPEG artifact removal. Useful for highly compressed JPEG images (e.g. those transmitted over wireless links).

[361] 4.18. DenoiseSaP – Salt and Pepper noise removal.

[362] 4.19. LocIllumCorrect – Correction of lighting non-uniformity. Useful for images of printed text.

[363] 4.20. PremHistEq – advanced histogram equalization for images with dynamic range problems.

[364] 4.21. ColorPaletteAdapt – Reduce the number of colors in an image using a fast algorithm. Useful for image file size reduction/adaptation to phone screens with a small number of colors.

[365] 4.22. FaceDetect – automatically detects a human face in a frontal facial image. Useful for capturing the most important part of an image for display on a limited size screen.

[366] 4.23. EyeDetect – automatically detects the eyes+nose section of a frontal facial image. Useful for capturing the most important part of an image for display on a limited size screen (e.g. Nokia Picture Message).

[367] 4.24. Add Text – Add formatted text to an image (with font selection).

[368] 4.25. Add Object – Add an object (hat, eyglasess etc.) to a photo.

[369] 4.26. Add Frame – Add a frame (several selections) to a photo.

[370] 4.27. Add Effect – artistic effects (warp, sphere, twirl etc.).

[371] 4.28. Embed Watermark – embed a watermark in an image/audio/video file.

[372] 4.29. Detect Watermark – fast detection of an existing watermark in an image/audio/video file.

[373] 4.30. Smart Compress – reduce the file size of the image/audio/video file to below a specified limit. Useful for reducing network bandwidth and for overcoming memory limitation in handsets.

[374] The MPS supports in a single product the complete range of processing requirements for the full spectrum of future MMSC infrastructure users:

[375] 1. The phone MMS user, composing and sending an MMS from a phone. In this scenario the primary need is for fast transcoding and automatic content type identification and processing. For example, images

taken by a user with a camera-phone need JPEG artifact removal, automatic contrast and color enhancement and face/eye detection for maximum utilization of target display screen size.

[376] 2. The Internet MMS user, composing advanced MMS messages from an internet-based interactive Multimedia Album. This user can play around with images and audio/video objects, add text/objects/frames to image, compose and use existing SMIL templates etc. Relevant functionality in the MPS includes support for interactive manipulation of images (adjust contrast, add formatted text, add a hat to a person in the image etc.), efficient storage of images (smart compress),

[377] 3. Advanced MMC scenarios, where a sequence of processing operations is performed on an MMS prior to sending – for example, detect watermark, block/report to billing system based on watermark info, compress audio component to reduce total MMS size while maintaining overall quality, convert video sequence to animations etc.

[378] 4. Content providers – these providers have large amounts of content with specific, detailed processing sequences based on their preferences/knowledge of the content characteristics. Such providers will utilize the more advanced options of functions such as Transcode, compress, color palette adaptation, embed watermark etc.

~~1.~~ Transcoding

[379] The main functionality of Transcode is to convert an image so it will fit into a target device while maintaining the best quality possible. In order to fit an image to a specific device, the main considerations are:

[380] 1. Resizing the image until it is small enough (in pixels) to fit the device.

[381] 2. Reducing the image's color/bit depths to the device capabilities.

[382] 3. Converting the image to the specified format – typically this format should be supported by the target device.

[383] 4. Ensuring that the resulting file size does not exceed the memory limitations of the device

[384]

[385] The algorithm used by Transcode can be divided into three main stages, according to the above criteria.

[386]

~~a)~~ Stage I: Resize-Fit

[387]

[388] In this stage the image is resized to fit the target device. For better quality, other image attributes (like bit depth) are not reduced yet (actually they may even be enhanced). Different variant of the resizing algorithm are used for different contentType values. Some parameters that may influence the result of this stage are:

[389] Device dimensions, scroll-size, maximal allowed pixels, etc.

[390] Source and target aspect ratio of pixels

[391] The choice whether to use just the physical screen or the full scrollable screen – this is controlled by a configuration parameter, but overriding it in the XML-request is possible (useScroll).

[392] The option to rotate the image by 90 degrees in order to get a larger view (rotatetobest) – the default value allows rotation for small-screen devices only (cell phones vs. PDAs, PCs). This may be changed in the configuration or the request itself.

~~b)~~ Stage II: Color Fit

[393]

[394] At this stage the image's bit depth and color space (i.e. color to gray) may be reduced in order to best fit a device. (For example, a color image with 24 bits of data per pixel may be reduced to a grayscale image with 2 bits of data per pixel in order to fit a screen that has only 4 gray levels).

[395] The image is run through a series of specially designed filters that maintain maximal image quality while reducing the bit depth.

[396] Specifying the contentType of the image can also control the behavior at this stage. For example, a lineart-type image is treated differently here, with filters that are designed to preserve as much detail as possible of lines and shapes, as opposed to a face/object image, in which the processing involves sharpening of facial features, or "scenery" photo-type image, in which the main point is to preserve color and brightness accuracy as much as possible.

~~c)~~ Stage III: Creating the output file

[397] When the image has reached its final size and depth, it must be converted to the format requested (after making sure it is supported by the target device). This stage could have been straightforward, but we must also make sure the file is small enough for the device's memory to handle. In some cases, after the file is created, it may be necessary to repeat the previous stages and create an even smaller image, until the file size itself is small enough.

~~d)~~ Other stages:

[398] Stage 0 in contentType = "document" consists of local thresholding.

[399] Reiteration of the process with stricter limitations if the output file size is too large.

~~2.~~ Watermarking

[400] Watermarking (WM) consists of embedding hidden information within media files/objects, which may be used as part of a digital rights management system (DRM) – for billing, copyright, content-blocking etc. The information content of the watermark in MPS is defined as a 19-digit numeric string, excluding 0 (i.e. 1<=WM<1019).

[401] Currently watermarking is supported for the formats jpeg, gif and png and is performed by hidden comments – for jpeg and png; these comments won't be visible through typical viewers. But it can also include watermarking of B/W at the image level, regardless of the format. The typical scenario for watermark usage is through devices that do not normally manipulate images, but may send images previously received from an MPS system without tracking information.

~~a)~~ Watermark functions:

[402] EmbedWatermark – This function is used to embed the watermark (numeric string). It can be used only when the specified output format is one the supported WM formats.

[403] DetectWatermark – This function detects the MPS watermark embedded in an image / media file. It is relevant only for WM-supported input formats. Note: The output of this function differs from typical MPS output – it is the watermark (or 'watermark not detected' message) and not an image.

[404] RemoveWatermark – This function is used to clear the watermark from an image. Note: This operation may happen as a side effect of most methods for some formats /implementations.

[405] It should be noted that currently the watermark functionality is designed mostly for demonstration and evaluation purposes. When integrated as part of a well-defined DRM system, watermarking functionality may include:

[406] Method X + PreserveWatermark: To maintain the identification of an image after transcoding / basic manipulation, an alias of the following combination may be used – DetectWatermark ->wm; method-X; EmbedWatermark (wm).

[407] Method X + ManipulateWatermark: Another possibility is that the output-watermark will have a different value than the input-watermark, either by applying some mathematical function to it, or by the some DRM component that will issue a new value and maintain a log of the relationship between these values.

~~3.~~ Overview of Image Processing Algorithms

[408] The system introduces a large number of image processing algorithms designed for:

[409] 1. Image adaptation and manipulation

[410] 2. Illumination correction

[411] 3. Noise reduction

[412] 4. Sharpening

[413] This grouping of methods is for the survey convenience only. The methods are simple enough to allow good definition of the parameters involved. Each method deals with common problems, relevant to image processing implementations. Still, the full collection of these methods does not allow dealing with complex problems, which are addressed by transcoding, premium and advanced packages. In complex scenarios it may be difficult to choose appropriate methods, for correcting the problem without introducing undesired side effects, which may degrade image quality to an undesired level.

~~4.~~ Common features

~~a)~~ Color treatment

[414] In the image manipulations, denoising and sharpening functions the colors are treated independently. This means that each method is well defined for grayscale images. Thus it is applied with the same parameters 3 times, once for every RGB channel. There is no essential difference in handling the indexed images vs. the continuous color/grayscale images. Such treatment of color channels is simple and intuitive. It allows better understanding and description of the parameters involved. More elaborate color space treatment shall be implemented in the context of premium package scenarios.

[415] In the illumination correction functions there are usually 2 modes of implementation with separateColors parameter acting as selector. When set separateColors=false, the method would handle all channels in a combined manner.

~~b)~~ Smoothing kernels

[416] Some algorithms included (noise reduction, sharpening etc.) use linear convolution with a pre-defined kernel as the main processing tool. The most common convolution is convolution with a simple Gaussian kernel. However, using convolution kernels with other shapes might improve the performance of the algorithms.

[417] The algorithms which are not very sensitive to the kernel parameters use Gaussian kernel with standard deviation automatically calculated from the effective radius.

[418] The algorithms which are sensitive to the kernel parameters use one of the following shapes:

[419] 1. rect – Rectangular shape is the most common, since it allows very fast computation. The problem with this kernel is its emphasis on diagonal edges, which are seldom present in the image.

[420] 2. diamond – This is a rectangular shape rotated by 45 degrees. The best feature of this shape is its ability to emphasize horizontal and vertical edges of man-made structures and geometrical objects. The other reason for emphasis on horizontal lines is the fact that they are hardly influenced by discretization process.

[421] 3. ellipse – Circular, or more generally elliptic kernel treats in the same way structures of every orientation. This is a more general-purpose kernel, used with natural images.

[422] 4. softEllipse – In ellipse the edges are hard-threshold: either 1 or 0. In softEllipse the edges can have a value between 0 and 1. This allows a better approximation of disc shape. This feature is suited for linear operations and may cause artifacts with non-linear filters (median, contrast stretch etc)

[423] 5. gauss – This stands for Gaussian filter, e.g. "gaus0707","gaus0505" and "gausAuto". The later two indexes stand for the standard deviation value of the Gaussian in each direction. The recommended setting "gausAuto" automatically calculates sigma based on the radius of effective coverage of the Gaussian. The Gaussian kernel allows graceful degradation of the pixel weight far from the center of the smoothing kernel. This feature is ideal for linear convolution.

[424] 6. kX – E.g. "k1" and "k2". Reference to bank of pre-defined kernels to be used in special scenarios. This is a 'premium' interface and there is no meaning to kernelWidth and kernelHeight.

[425] It is possible to use prolonged kernels to emphasize the horizontal and vertical edges. For this purpose separate parameters are defined for kernelWidth and kernelHeight rather than radius.

[426] Fine-tuning kernels for images and algorithms can be a tedious task, therefore some basic recommendations are given where possible.

~~5.~~ Image adaptation and manipulation

[427] The elementary image processing operations deal with image size and orientation. These functions, namely Rotate, Resize, Flip and Crop, are available in every basic image manipulation package. Threshold, Compress (with GIF output) and Invert methods are used to adapt the image to the display color palette and minimize the use of target device memory space. Resize and Threshold provide advanced modes - which perform more sophisticated processing.

[428] The methods in the basic image manipulation package can be optimized for speed, and can include platform specific speed-ups in all platforms (Intel, Solaris, etc).

[429] In addition to providing an image manipulation package, these functions enable advanced compression and building your "home-made transcode". Typical variations of such experimentations with transcoding consist of a sequence such as crop->resize->compress, or crop=> (optional) rotate=> resize=> quantization/threshold - to adapt large images to small displays. Crop allows spending all of the limited screen resources on the main object; resize minimizes the information contained in the picture and compression/quantization discards less-important information. Rotation is used to use the display dimensions and aspect ratio as best as possible. On some media the image has to be inversed prior to display (e.g. scanned negatives).

~~a)~~ Color palette adaptation

[430] ColorPaletteAdapt fits the image to a limited palette. This is useful either when the device or file-format has a limited color capability or when file size is an issue. Once the palette is defined, each pixel is assigned a value from it: this is done either by assigning each pixel the nearest value, or dithering – a method which increases color resolution at expense of spatial resolution. As default, dither is used when the specified number of output colors is small, but the user may explicitly specify whether dithering should be used. Dithering is not recommended when output file size is a major issue, but is recommended when the device color capability is the issue. Note that ColorPaletteAdapt with paletteName="B/W" is equivalent to threshold with the default parameter, for B/W adaptation one may prefer using advanced modes of Threshold.

~~b)~~ Threshold

[431] Threshold converts a grayscale image into a discrete B/W image. It may used as part of other more complex conversion operations (e.g. Transcode), and can serve for artistic effects or image combination effects. For example: converting a formatted text/textured text image into B/W before sending to a B/W screen, reducing the color content of a single layer (e.g. object for combining in an image) so it will not add to the color palette of an image etc. It allows explicitly controlling the threshold level, automatically finding the optimal global threshold, or applying a local threshold (mode localV2 is usually inferior to local).

~~c)~~ Compress

[432] The method compress attempts to reduce file size without changing the image size. It may achieve this goal by more efficient coding, reducing colors and losing some details. In some case it may follow an operation

intended to reduce the image size. It is wise to apply compress in combination with an efficient image format (i.e. jpeg / jpeg2000 for storage, gif for most devices)

[433] This function has a different implementation according to the requested output type. The problem of file size is important in several aspects: Most of the currently available phones / hand-held devices have limited capacity, both for single file and in total. Bandwidth in cellular networks may also be an issue. In addition, in messaging system storing many millions of MMS messages, the typical file size becomes an issue to consider too. On the other hand most images entering the system may not be limited to the few-Kbytes range suitable for hand-held devices.

[434] The palette and processing power limitations of currently available hand-held devices makes the compression utility especially relevant for image/gif output mime type.

[435] In this case compress activates adaptive quantization procedure, which provides for a clear image with minimally reduced color palette. Detail reduction, image resizing and cropping are not supported by the compress method and require dedicated requests.

[436]

[437] The implementation of the algorithm is based on adaptive reduction of the color palette and smoothing for GIF/PNG images, and on JPEG DCT quantization table variation and smoothing for JPEG images. The parameters are changed iteratively until the maximum quality setting with a file size under the limit is reached.

[438] It is important to know that if the target file size is too small, the algorithm will return an error message. This reflects the fact that even under a color reduction to a bitonal image the file size under the given compression method was too large. In this case the image should first be resized, then compressed.

[439]

[440] The parameter available for this function is maxSize, which is the maximal allowed image size in bytes. For a CIF sized image, to be displayed on a typical hand-held device, some recommended sizes are:

[441]

[442] Image /Device type MaxSize (bytes)

[443] VGA image – storage Up-to 50000

[444] Lineart/MMS message 4000

[445] Lineart/WAP phone 1400-2000

[446] T68/WAP client 2300

[447] Natural image/PDA with MMS/email client 8000

[448]

~~d)~~ Invert

[449] Some image sources (e.g. scanned negatives) and output devices require image inversion. The inversion is performed for each color channel separately, so that yellow is transformed to blue, white to black etc. This is a simple function so it does not require any parameters. It is most useful for e.g. synthetic images displayed on low contrast screens.

~~e)~~ Rotate

[450] This is a standard implementation of image rotation. The parameter specifying the amount of rotation counter-clockwise in degrees is mandatory. Values out of 0-360 range are corrected by the algorithms, so that -90, 270 and 630 degrees rotation have the same effect.

[451] For angles other than 0,90,180,270 the output pixels of the original image are not a rectangular image, this poses a choice of what rectangular image should be returned and what should be the values of the pixels not present in the input image (in 90,270 the size changes - width <-> height - but there is no dilemma defining the output image). The parameter mode determines the output size: mode=full returns the bounding block image of the rotated input image, mode=crop crops the rotated image to the size of the original image (with the same center). The portion of the output image, not present in the input is always padded with a black background. For rotation by multiples of 90deg both modes provide the same output.

[452] The interpolate parameter allows to choose the interpolation technique. interpolate=bilinear is usually a good choice. It is computationally efficient and does not introduce large artifacts. Selecting interpolate=bicubic provides for a more sharp and accurate image at cost of computational efficiency and ringing artifacts. interpolate=nearest selection is most efficient computationally and does not alter image palette. However it may introduce some aliasing artifacts. This parameter is also ignored for multiples of 90deg.

~~f)~~ Resize

[453] This speed-optimized function serves to change the image size. It can be used to fit an image into a small phone screen, or to reduce an image size prior to compression and storage. For example, an incoming 3 mega-pixel image from a high-quality digital camera may be resized to VGA (640 by 480) size prior to JPEG compression and storage, in order to reduce storage space requirements.

[454] The mode parameter selects the interpolation algorithm. Beside the usual bilinear, bicubic and nearest methods, proprietary methods are supported to provide for optimal performance with various image types and target devices.

~~g)~~ Flip

[455] Flip the image horizontally (vertical=false) or vertically (vertical=true). For an upside-down one should call this method twice, each with a different parameter value (or call Rotate 180 degrees).

~~h)~~ Crop

[456] Crop may be used when the final image size is limited and the more significant details are concentrated in a limited region of the image. Cropping most of the background allows applying a more moderate resize. After the application of crop method a rectangular area 'cut' from the original image is returned. Crop's interface is the following

[457] top -the upper bound of the image, range: 1-ImageHeight

[458] bottom -the bottom bound of the image: top-ImageHeight

[459] left -the left bound of the image: 1-ImageWidth

[460] right -the right bound of the image: left-ImageHeight

[461] The coordinates start from the top-left corner of the image with coordinate (1,1), rather than (0,0) used in some other commercial software. The rectangle specified by the four parameters has to have a positive area, so left <= right, top<=bottom. The edges are included in the rectangle.

~~6.~~ Illumination correction and color manipulations

[462] The illumination correction is one of the more difficult problems in image processing. There are many ways to correct for improper illumination/ detector problems. Basic solutions work only on a small range of imaging situations. The methods given below are just the most simple and intuitive tools, while the premium package contains more complex and elaborate algorithms to deal with the problem.

[463] The AutoLevel method with empty radius parameter is perhaps the most powerful algorithm available in the basic package.

~~a)~~ Darken

[464] This method darkens the image, and has two modes:

[465] If intensity is not specified, the function performs a darkening which may be described as the dark half of AutoLevel.

[466] If intensity is set, the image is darkened by the specified amount (intensity : [range: 0-1 (0- do nothing, 1-maximal)]). When intensity is 1, all mid levels become black and only colors white (or brightest level in channel X) remains as it was. Darken with intensity X is equivalent to Adjust levels with contrast=0, brightness = -X.

~~b)~~ Brighten

[467] This method brightens the image, and has two modes:

[468] If intensity is not specified, the function performs a brightening which may be described as the brighthalf of AutoLevel.

[469] If intensity is set, the image is brightened by the specified amount (intensity : [range: 0-1 (0- do nothing, 1-maximal)]). Brighten with intensity X is equivalent to Adjust levels with contrast=0, brightness = X.

~~c)~~ AdjustLevels

[470] This method gives the user full control and responsibility of the output. Both parameters are mandatory.

[471] brightness : [range: -1 – +1 (-1 = black, +1 = white), recommended range: -.3 - +.3]

[472] contrast : [range: -1 – +1: (-1=monotonic image, +1=pure color image), recommended range: -.3 - +.3]

[473] For positive contrast values, first the brightness is adjusted and then the contrast. For negative values, contrast is adjusted first. The effects of both these operations is accumulated, so the maximal effect when the contrast is positive is the sum of the contrast value and the absolute value of brightness. This sum should not exceed 1, or be too close to it (i.e. contrast=brightness=0.6 will result in a white image, just like brightness=1).

~~d)~~ AutoLevel

[474] This is the primary function for illumination correction included in the basic package. The global AutoLevel (empty radius parameter) maximizes

image contrast, discarding the outliers of very high and very low intensity. This is especially useful for images taken in the haze, rain etc.

[475] The local AutoLevel is activated, setting some positive radius parameter. The recommended radius values are in range [10-30]. For small radius values the image appears grainy. Unlike the global AutoLevel, the local AutoLevel stretches the contrast w/o outlier detection. This effect is achieved if used after DenoiseSaP.

[476] The separateColors=true setting allows illumination/color correction as well as luminance correction.

[477]

~~e)~~—ColorBalance

[478] This function tries to produce truly colorful images, by increasing the contrast in hue domain. Thus a white object on a blue background will appear yellow. This method is similar to the correct illuminant effect in vision. On most natural images the effect is very small. Setting mode=1 is mandatory. The level can vary in range [0-1] with default 0.5. Both parameters are mandatory.

~~f)~~—WhiteBalance

[479] This function calculates the mean value of the gray pixels in each channel and brings it to 0.5. The pixel is reported as gray if its color in all 3 channels obeys abs(pixel_value- 0.5)<tolerance. The tolerance is a user-supplied parameter with default value 0.15. The color correction is a gamma correction, with automatically calculated parameters for each channel.

~~g)~~—ColorVariations

[480] This is a two-stage function. In the first stage the gamma of each color channel (e.g. red, green, blue) is changed according to intensity value between –1 (remove color) to +1 (saturate color). In the second stage the saturation of the color is changed by changing the Euclidian distance between each channel value and gray image value, such that –1 stands for grayscale and 1 stands for saturated colors.

[481] For a simple cmos based camera such as the Communicam it is usually recommended to put

[482] red = .2

[483] green = .2

[484] blue = -.1

[485] saturation = .3

~~h)~~—HistEqualize

[486] This method performs Histogram equalization (no parameters). The resulting image has a uniform histogram (as much as possible considering the input color distribution). This is a common solution illumination correction, but it has side effects, such as eliminating the real color distribution of the image (e.g. adaptive thresholding of the result of histogram equalization, is likely to have poor results).

~~i)~~—PremHistEq

[487] Unlike the HistEqualize, PremHistEq trades off the speed and simplicity for the flexibility of operation. It has a large set of parameters and modes of operation which have different effects.

[488] P-law histogram equalization allows a trade-off between simple histogram equalization (pval=0), no effect at all (pval=1), dominant modes emphasize (pval>1) and dominant modes destruction (pval<1). The recommended values are

[489] Pval Scenario

[490] 0.3 Histogram equalization of moderate intensity

[491] 0.7 Histogram equalization of small intensity

[492] 1.5 Posterization

[493] -0.2 Histogram equalization of very large intensity

[494] Local histogram equalization allows histogram equalization on blocks of limited size. The recommended size of the block is in the range 0.2 – 0.6 of the image size.

[495] Number of histogram bins has quantization effect. For pval<0 it is recommended to use at least 32-64 bins. The algorithm uses linear interpolation between bins.

~~j)~~ LocIllumCorrect

[496] This method performs local illumination correction and has a large amount of sub-methods chosen by correctionType. Other methods which locally correct the illumination level are AutoLevel (local) and, for binary output, the local mode of Threshold. This procedure is effective for non-uniformly lighted handwritten and printed text as preprocessing to advanced applications, such as OCR and feature detection, but it may sometimes degrade the visual quality of the image as perceived by humans. Some safety mechanisms were introduced to limit the visual degradation of the image. One of this mechanism is setting separateColors=false to preserve the original hue of the image.

[497] The LocIllumCorrect can produce unexpected results with images with very limited color palette (16 colors and less).

~~k)~~ GlobIllumCorrect

[498] The global/block-wise illumination correction allows automatic correction of image curves with the following sub-methods chosen by correctionType:

[499] gamma – gamma correction, so that the mean value of the image is .5 (gray).

[500] curve – a variation of gamma correction with highlights and shadows subjected to separate gamma values.

[501] contrast – synonym for AutoLevel. The addition functionality is bloc-wise processing.

[502] histEq – synonym for PremHistEq with different interface (number of bins and power is selected automatically).

[503] The correction is global, unless blkHeight and blkWidth are both set. The recommended block size is 64x64 or 128x128. The blocks overlay, so that their borders are virtually invisible for block size larger than 32x32. The separateColors parameter allows to select color channel treatment. Usually separateColors=true provides for desired color correction.

[504] The GlobIllumCorrect can produce unexpected results with non-photographic images (e.g. lineart) and images with very limited color palette (16 colors and less).

~~I)~~ Enhance

[505] This function performs a selected combination of methods based on enhanceType parameter.

[506] If the enhanceType is empty the function performs mild color balance + contrast enhancement. It can be used safely on various input images and should improve many of them. The effect is similar to ColorBalance followed by AutoLevel (global).

[507] If enhanceType=communicam, batch processing is executed using:

[508] o ArtifactRemove => Deblur => PremHistEq => Crop => ColorVariations => GlobIllumCorrect. This processing is essentially acceptable for various JPEG input images, but most suitable for communcam output images.

~~7.~~ Noise reduction

[509] The two most common types of noise treated by the basic package are:

[510] 1. White Gaussian noise / speckle noise

[511] 2. Salt and Pepper (S&P) noise

[512] White Gaussian noise appears as an intrinsic part of the cheap camera detectors, especially in low illumination conditions – it is inaccuracy in the pixel values – for many pixels. This is the most common type of noise, which appears on most of the images.

[513] The S&P noise, is a small number of pixels having big "errors" in their intensity levels. It appears as a result of interlacing/aliasing in the detector, faulty detector, sharpening of degraded images, communication problems, poor JPEG compression, scanning of analog photos. This type of noise is more rare and easier to treat than the Gaussian noise.

[514] Since the noise appears independently in each color channel, the noise reduction procedures are independently applied to the color channels.

[515] The noise types defined in this procedure do not have clear meaning in line-art images, therefore a natural image source is assumed. For indexed images a rich color palette is required.

~~a)~~ Smoothing

[516] The most common and simple way to deal with noisy, over-sharpened images and compression artifacts is smoothing. Smoothing is performed by a simple procedure of convolution between the original image and a Gaussian kernel.

[517] The output of this method is a smooth image, where the degree of smoothness increases with the optional intensity and radius parameters.

[518] Since there is no common scenario for the use of smoothing method, there are no clear recommendations about the smoothing parameters. For medium smoothing which does not degrade image significantly we can recommend intensity=0.5, radius=3.

~~b)~~ DenoiseSpeckle

[519] This is essentially an empiric Wiener filter. It essentially uses algorithm developed by Lee (1980) with minor modifications. The main modifications

include a bank of smoothing kernels, the overfilter parameter and regularization process.

[520] This de-noising procedure is essentially a local adaptive smoothing procedure, where the overfilter parameter plays the role of smoothing intensity, and the smoothing kernel selection plays the role of radius in the smoothing method.

~~c)—~~DenoiseSaP

[521] This is a standard 2-stage procedure of outlier detection followed by their replacement with local median. Changing the threshold influences the detection rate. Increasing the threshold will allow under-smoothing, while decreasing the threshold will allow over-smoothing. It is recommended to work with kernels of 9-25 pixels.

~~8.—~~Sharpening

[522] On most amateur photos the image is not sharp enough. The common reasons are: bad focus, motion blur, JPEG compression. To increase the sharpness of the image one can use a sharpening procedure.

~~a)—~~Sharpen

[523] This function implements a standard unsharp masking procedure (which is equivalent to smoothing with negative intensity) if edges=false. In this setting the function increases the noise in the image, which is usually not recommended with originally noisy images. Setting edges=true will result in sharpening only over the edges, which is a preferable mode of operation. In this mode the radius parameter is ignored. The level of sharpening can be controlled by the intensity parameter.

~~a)—~~ ArtifactRemove

[524] This function deals with heavy JPEG artifacts only. The artifacts are listed below:

[525] 1. Blocking: various sharp borders between 8x8 tiles

[526] 2. Ringing: high frequency moir' noise in 8x8 tiles

[527] 3. Color spill: color channels not coinciding with luminance channel

[528] 4. Blur: high frequency edge blur

[529] The ArtifactRemove method with deJPGtype=detail settings does not degrade the image visual quality. This is a general-purpose tool, which can be used for any JPEG input. However, for any specific problem it is recommended to use the dedicated method instead.

[530]

~~b)—~~Deblur

[531] Most of the digital photos taken by low-quality cameras appear to be out of focus or blurred. Simple sharpening does not solve this problem completely, especially in the noisy environment. Some off-the-shelf products (e.g. Extensis Intellehance) suggest blind deconvolution with adaptively selected kernel. The kernel is selected, so that maximal sharpening is achieved with minimal ringing artifact. While this method is implemented deblurType=defocus, it is not recommended in most cases. In the presence of noise and JPEG artifacts, the proprietary deblurType=premiumSharpen method proves to be more consistent and does not degrade image visual quality. The main difference between

Sharpen and Deblur (premiumSharpen) is the use multiple kernels and homogenization as a part of sharpening procedure. For more sharpness one can use deblurType=premiumSharpenMore.

~~10.~~ Functional description of AudioTranscode

[532] The support of multiple input and output format, optimized for the specific device, is as important in sound processing as it is in image processing. The main sources of audio are MP3 and WAV files, available on most PCs, disks and internet sites. These formats allow very rich sound at the expense of the disk space. The transmission bandwidth and memory of the hand-held devices is very limited. The audio capabilities of these devices, don't always make use of the richness of the input formats (e.g. stereo) More efficient compression techniques (e.g. GSMAMR) have been defined in the context of MMS for playing/recording audio on these devices. The audio transcoding process performs the following tasks:

[533] 1. Down-samples and compresses rich audio files (MP3,WAV => GSMAMR)

[534] 2. Masks compression artifacts and allows to play compressed files on PC (GSMAMR=>WAV, effect=mask)

[535] 3. Converts GSMAMR files from abundant to minimal compression rates and vice-versa (GSMAMR=>GSMAMR)

[536] 4. Supports Unix audio standards (AU to anything and anything to AU)

[537] 5. Adds effects to audio files (via the effect string) for audio quality improvement without issuing a separate request.

[538] 6. Automatically selects the conversion method via the mime type.

[539] 7. Performs ringtone conversion between Nokia Ringtones (part of the NMS standard) and iMelodies, TDD polyphonic tones (part of EMS standard and extensions to it respectively).

~~IV.G.~~ Other specific enhancements

~~1.~~ Using the MMS Box as a WAP Site

[540] WAP terminals have a built-in WAP browser. It is possible to go to a Web site with the terminal, and call down relevant information. The server will process the information called to optimize it for display on the terminal, and the processed information will then be transmitted to the terminal for display. This information may or may not be processed further by the terminal or by the server, according the user's request. Information which has been processed (either once or twice) may then be stored, in the terminal, or at the server, or at another information storage place specified by the user. Transmission to and from the terminal may be by wireless or wireline communication.

~~2.~~ Converting a WBMP Into a Picture Message

[541] WBMP is the WAP protocol for graphics. Images on the terminal may be displayed in PM format, not WBMP. The server may receive a WBMP image, convert it into the PM format, and transmit the message for display on the terminal. This conversion is new because the protocols WBMP and PM are both new, and therefore the conversion has not been performed previously.

[542] Converting a WMBP Image into an EMS Picture Message

[543] This innovation is exactly the same as converting a WBMP into a picture message described above, except that the target format for display is EMS rather than PM. It will be appreciated that images may be converted into any number of picture formats, be it PM, EMS, or a different picture format to be devised in the future, such as Smart Messaging. The algorithms in the server make this transcoding possible. Again, this is new because the protocols, WBMP and EMS, are both new, and therefore the conversion has not been performed previously.

~~3.~~ Human Face Recognition and Display

[544] Refer to Figure 1 attached to and incorporated into this application. In the picture, the woman's face may be recognized by algorithms defined in prior art. The invention includes innovative algorithms resident in the server which allow the server to process the relevant part of the picture, in this case the woman's face, for display on the terminal. There are three types of algorithms. The first is orientation. The face is oriented vertically, which means that the vertical dimension of the relevant part of the picture is greater than the horizontal orientation. Some terminals have display screens that are wider than they are tall. To capture the full image on one screen would reqire a reorientation of the woman's face from verticl to horizontal. The server knows the display characteristics of the terminal, and will perform this orientation.

[545] The second kind of algorithm is "resizing". Terminal displays are generally smaller, often much smaller, than the source image. The server will know these characteristics, and will accordingly resize the picture for display on the terminal.

[546] The third group of algorithms is those which will reproduce the image on the terminal's display, while maintaining the integrity and quality of the image as much as possible. The need for these algorithms arises from the small display screen, or from the inherently lower resolution of the terminal display, or from other reasons. The server will know the characteristics of the terminal display, and will apply the correct algorithms for maximum effect. Examples of such algorithms include enhancement, dithering, and histogram correction.

[547] The application of any or all of these algorithms to handset displays is innovative. The use of prior art face recognition as part of the system and method described herein is also innovative.

~~a)~~ Face Binarization

~~(1)~~ Faces to BMP

[548] FIG.18 shoes a block diagram explaining the procedure.

~~(2)~~ Envelope:

[549] Image is more or less frontal, eyes should be visible, and illumination variations should not be too extreme. Constraints are set both by face detection requirements and by binarization requirements. Size of face in image should be sufficiently big.

~~(3)~~ Components:

[550] a. Face detection, eyes detection:

[551] Preferably, this would be a face detection SDK (e,g, trueFace, FaceIt).

[552] b. Illumination correction (to the extent possible).

[553] c. Facial features emphasizing:

[554] Increasing contrast – brightening the skin, darkening features (eyes, lips, eyebrows, etc.) and hair.

[555] Sharpening.

[556] d. Optimal resizing – undo the blur.

[557] e. Binarization with dithering.

~~(4)~~ Faces to Picture message:

[558] Output: The eyes strip.

[559] Additional components: eye detection.

[560] This is especially useful when one has to display a part of the human face on screens that dictate a wide and short frame size – e.g. many phones have an aspect ratio of 2:1 or more in the width/height of the display. In addition, the PM (picture message) format of Nokia Smart messaging dictates that images are at most 72 pixels wide by 28 pixels high. See examples in attached figure 26 showing the extraction of the eye region from an image and then converting it to a picture message.

~~4.~~ Combination of Histogram Correction and Dithering

[561] A histogram in the current context is the process by which the various pixel values in a grey level image are distributed on a frequency chart, from pure white through various shades of grey to pure black. Histogram correction is the process by which some of these values, but not all, are lightened or darkened, but even those values affected are changed to different degrees. Dithering is the translation of grey level images to black & white by the correct combination of the black and white pixels to simulate grey in the eye of the user. The use of dithering for small screens, such as those on the terminal displays discussed here, is entirely new. Histogram correction, even for small screens, is not new. However, the use of histogram correction in the method and system described herein is new. Further, there is a technique by which histogram correction is applied first, and then dithering, to transcode a very high quality image onto a terminal display that protrays only black & white images. This technique is entirely new, and is part of this invention.

~~5.~~ Combination of Floyd Steinberg Dithering and Random Permutation

[562] Floyd Steinberg dithering is a well-known dithering algorithm in which error diffusion methods are used to create visually appealing dithering with relatively few fixed repeating patterns. Random permutation is a technique by which a few random black pixels are changed to white and a few random white pixels are changed to black. Random permutation is used to avoid "periodicity", which is a situation in which there are appear to be very dramatic changes in shading from one part of a picture to an adjacent part of the picture. This problem is particularly prevalent when large pictures are compressed into a smaller area such as a small display terminal. Random permutation softens the effect of these changes. Floyd Steinberg dithering is part of prior art, as is random permutation. However, the combination of first Floyd Steinberg dithering and then random permutation is new, and is part of this invention. Further, the addition of this combination followed by transcoding to WBMP, EMS, or PM, as the case may be the particular target terminal, is entirely new, and is part of this invention.

~~6.~~ Correction for Non-Square Pixels in the Target Display

[563] Not all terminals have square pixel displays; sometimes the pixels are rectangular. If that is the case, then the server may need to convert say a 100 x 100 square pixel picture into say 80 x 100 rectangular pixel display on the target terminal. The server will know the chacteristics

of the display terminal, and will perform the required correction; that technique is new. Further, the addition of this technique with other techniques described here, and the transmission of the processed information, is entirely new, and is part of the method and system described herein.

~~7.~~ Transcoding of Color and Shade in Images

[564] Images generally in one of three types of color or shading, which are are color (green, blue, and red, or other permutations), grey level, and black & white. Terminal displays today are generally black & white only, although grey level is being introduced, color displays are planned for the future. The server will know the characteristics of the display terminal, and will be able to transcode source images into a display format suitable for that terminal. Note that it is possible to transcode down, or to transcode up from white & black to grey level, but it is not possible to transcode into color. That is, the various possible permutations are color to grey level, color to black & white, grey level to black & white, or black & white to grey level. However, grey level or black & white to color is not possible. Also, needless to say, it is possible to transcode at the same level, such as black & white to black & white, although different algorithms will be employed to to maximize integrity and quality of the image on the terminal display.

~~8.~~ Panorama Imaging on a Terminal

[565] Refer to Figure 2, attached to and incorporated into this application. Panorama imaging requires that multiple pictures of a scene be taken, and then various images be "stitched" together in order to create one continuous scene. [Note: Panorama imaging, and the various algorithms employed to make it possible, is the subject of a separate patent application by the current assignee, UCnGo, Inc.] By the nature of the limited size of the terminal display, the entire picture cannot be displayed on one screen. Scrolling is required. In addition, re-orientation of the image may also be required, as demonstrated in Figure 2. It should be noted that long text messages, which may not be displayed on one screen, may also be formatted for scrolling, and again the scrolling may be either vertical or horizontal, depending on the type of terminal display and the nature of the text.

~~9.~~ Embedding a WAP Link in an SMS Message

[566] This in itself is not new, since it is part of the prior art. However, it is new as part of the method and system described herein, particularly where the SMS link message serves as a method to deliver multimedia content to the user of a WAP phone.

~~10.~~ OTA Bookmarks and Enrollment

[567] "OTA" means "Over the Air", and is a short expression for real time action, in this case via a radio system. To do personalized bookmarks and home pages OTA is part of the prior art. Each user is assigned a specific URL ("Uniform Resource Locator"), with a common beginning string and some additional user-specific string (e.g., http://wap.ucngo.com/mmsservice/userboxes/userid=45FC9E0, where the bold part is user-specific part). In this way, different users receive

different bookmarks (URLs), and when a user operates a specific WAP browser, the UCnGo server receives an HTTP request with a URL unique to the given user, and the server will know what information to display based on the specific URL. To give each user his or her own URL, is also part of the prior art, but only in a wireline environment. To give a URL to a wireless subscriber is new. To combine ULR with OTA for personalized bookmarks, personalized home pages, URL links, enrollment in various services, and other services, is new, and is part of this invention. This combination requires both the application of database technology and the algorithms defined in this application.

~~11.~~ Correcting for Inversion of Pixels on Target Display

[568] In some terminals, the display has been intentionally inverted. That is, if a 0 bit is usually white, and a 1 bit usually black, in an inverted terminal the 0 bit appears as black and the 1 bit appears as white. Therefore, if any MMS message at all is sent, the display in an inverted terminal will show the message as a negative of the original. A Timeport phone, by Motorola, Inc., is an example of such an inverted display. This inversion is not a fundamental problem, as long as the server knows the terminal characteristics, and can correct for the inversion. The server in this application does know the terminal characteristics, and will correct for the inversion before the message is transmitted.

~~12.~~ Transcoding of Text or Numerics into a Picture

[569] The Invention will transcode text or numbers into a picture, in WBMP, PM, or EMS format, as required by the target display. This is new.

~~13.~~ Use of Modules Such as OCR and ICR to Identify and Process Text and Images

[570] The server uses OCR and ICR (Intelligent Character Recognition) to identify which parts of an image are text. Reference is now made to Figure 3, incorporated into this application. The first step in processing an image is the processing of the image into parcels such as text and drawing. Different processing techniques are then applied to each type of parcel. Rules will be applied, such as "Ignore grey level information" because the image may be in black & white, or "Maintain line solidity". Without the parsing and application of techniques, the image will be reproduced on the terminal in a manner similar to what is written as "Naï ve Transcoding" on Figure 3. With the invention, the "Optimal Adaptation" level is achieved. This process is part of the invention.

~~14.~~ Flexibility Resizing

[571] As a specific case, "flexible resizing" is a technique by which different parcels are resized differently; for example, text may be resized as little as possible to maintain legibility, whereas an image, such as that portrayed in Figure 3, may have a greater degree of reduction, since only recognizability, not legibility, is required. Flexible resizing is also part of the invention.

~~15.~~ Adaptive Classification Engine for Smart Resizing

[572] A variation of flexible resizing is where the decision of flexible resizing is generated not solely by recognition algorithms, but rather by recognition algorithms in combination with parsed samples. The first step of the procedure is that various sample images are fed into the server's

database. These images have already been parsed, and individual parcels have been identified as requiring different processing algorithms, in various orders of operation. The parsing, algorithms, and order of operation for the algorithms, have been tested by both theory and trial & error, and have found to produce optimal results. When a new image is then received by the server, the image can be parsed, the parsed parcels will then be compared to the database of parsed parcels, and the classification engine will then choose, on the basis of the samples and the target image, which algorithms and which combination of algorithms to apply to each parcel. This classification is "adaptive" in that in changes either with the addition of samples, and/or with feedback from the results of processing on real images. The adaptive classification engine is like a learning machine that applies rules and improves its own performance over time. The concept of a learning machine by itself is prior art. An adaptive classification engine for smart resizing of MMS messages is entirely new, and is part of this invention.

~~16.~~ Vectorizing and Processing Charts and Cartoons

[573] The processing of charts and cartoons is similar to the description of Use of Modules Such as OCR and ICR to Identify and Process Text and Images (see above). It is portrayed in Figure 4, hereby incorporated into the application. That is, the image will be parsed to determine the various parcels to be processed. Then rules will be applied. For display of a standard size chart on the small display of a terminal, for example, superfluous information such as a series of values on the axes will be removed. Also, a rule such as "remove horizontals" may be applied, since the addition of the horizontals to the small display screen will make the graph almost unreadable. For the graph itself, a rule such as "maintain line solidity" may be applied. The entire image will then be resized to the small display. Vectorizing algorithms are prior art. For example, Adobe Streamline Software uses this technique. The technique has not been applied to small screens such as the terminals discussed herein, and that is new. The combination of that technique with resizing and the other operations described herein are part of the method and system of this invention.

[574]

[575] This is a further explanation of the vectorizing and processing of charts:

~~a)~~ Charts to WBMP

~~(1)~~ ~~Envelope~~

[576] Standard charts – graphs of a continuous one-dimensional function, e.g. stock charts and other temporal variables. (A better solution can be provided more easily for a known source with known format).

~~(2)~~ ~~Components~~

[577] a. Identification and separation of graphics to content layers: graph, grid and boundaries, graph label, range text (numbers on both scales), background. b. gnore clutter (background, grid, possibly grid boundaries).

[578] c. Handle range: Ignore most values; maintain min, max at required size.
[579] If possible: safely resize. Use Ocr if needed/possible.
[580] d. Handle graph label:
[581] Long text: ignore, or ocr and return as text.
[582] Short text (e.g. stock symbol/name): preserve label, but move it as needed.
[583] Problem: range, label may not always be properly identified/handled for unknown format.
[584] a. Resize graph:
[585] Determine the maximal allowed size of the graph (after labels "waste"): With maximum scroll; with graph fully in screen; with graph and label fully in screen.
[586] Determine which size to use.
[587] Resize graph maintaining line continuity, and without thickening lines without necessity (consider graph as b/w).
[588] Aspect ratio does not necessarily need to be maintained.

~~(3)~~ Block diagram:

[589] FIG.18 shows a block diagram for the procedure

~~b)~~ Charts to Picture message

~~(1)~~ Components – changes from wbmp:

[590] a. Grid boundaries ignored (or just marks); b. Handling range: Ignore horizontal range; c. Range and labels: Use Ocr if possible (return: graph + "MSFT, range 80-115"); d. Resizing graph is the same (with different parameters).

~~17.~~ Conversion Algorithms s

~~a)~~ Cartoon

~~(1)~~ Cartoon to WBMP

~~(a)~~ *Envelope:*

[591] Content for the system is b/w line art. The size and amount of text should allow applying moderate resizing and fit, or recognizing through OCR. Typical samples are available at "K:\QA\Image_Banks\Comics\sliced comics\b&w".

~~(b)~~ *Components:*

[592] a. Correct conversion of generic content within the one-block envelope.
[593] b. Generic block identification and splitting
[594] c. Handling text:
[595] i. Identifying text.
[596] ii. Identifying whether standard conversion would be ok.
[597] iii. More moderate resizing + word reordering
[598] Problems:
[599] This may not always fit. Do we split it to multiple images?
[600] Identifying optimal resizing.
[601] iv. Specialized resizing – to allow legibility after resizing.
[602] v. Recognizing with Ocr:
[603] Evaluate and tune use of both scansoft and ParaScript (cartoon text is closer to handprint)

[604] Appropriate preprocessing.
[605] Problems:
[606] Cartoon text is small and not neatly printed – Ocr may fail.
[607] Handling non-English text.
[608] vi. Support getting text as input.
[609] d. Recognizing and cleaning non-relevant text

~~(c)~~ *Block Diagrams*

[610] FIG.19 shows a Block diagram without OCR.
[611] FIG.20 shows a Block diagram with OCR:

~~(2)~~ Cartoon to Picture SMS

[612] All limitations and stages for wbmp apply.
[613] Further envelope constraints:
[614] Text must be Ocr-able or given as input.
[615] Text limited to 60 characters.
[616] Image part content must fit in 28 pixel (height).

~~18.~~ An Example of a Combination of Techniques: Dithering, and then Adapting Photographs (with different image material) to WMBP, EMS, and/or PM, format:

~~a)~~ "Generic" photo binarization

~~(1)~~ Photo to WBMP

[617] Typical scenes to be handled: car(s), profile of person, full body, multiple faces/people, skyline, signs, trees, etc.
[618] Desired output is the silhouette of the object(s) in the image, an identifiable binarization of the scene.
[619] FIG.20 shows a Block diagram for this procedure

~~(a)~~ *Components:*

[620] a. Emphasizing boundaries, decreasing intra-surface changes (illumination, etc.).
[621] b. Resizing
[622] c. Identifying silhouettes.
[623] d. Brightening background
[624] e. Appropriate dithering within objects – when needed (only big surfaces).
[625] f. Combining the silhouettes with the surfaces.
[626] A data set for tuning and evaluation should be collected from current image banks + web.

~~(2)~~ Photos to Picture message:

[627] Problem: most scenes can't be converted to 28 pixels. It is usually difficult to identify important parts of the scene
[628] Additional components:
[629] "Central object" detection.
[630] When that is not possible – additional resizing.

~~*IV.H.*~~ *VAS extended functionality*

[631] It is expected that a large portion of MMS traffic will be generated by VAS application such as: photo albums, game sites, news sites etc. The UCnGO MCS provides special functionality for such services:
[632] 1. Watermark embedding/detection for digital rights management (DRM) – in order to track/block/limit the distribution of copyrighted content, a mechanism is provided to embed a watermark in any media

object. This watermark does not change the image/sound of the media object in a perceptible way. It should be noted that this method does not interfere with other DRM mechanisms such as those provided already by Nokia (by special tags in WAP pages) and Ericsson (tags in content descriptor). It provides an additional mechanism which is independent of external tags.

[633] 2. Content Based Transcoding – the conversion of visual information (images, animations and video) to displays with limited color range, memory and resolution is enhanced by knowing the type of the visual content. For example, the optimal conversion algorithm of a chart into a B/W display is different from the optimal algorithm for a natural photograph. See figure 4 for examples.

[634] 3. Layered graphics input – the vast majority of professional graphic and photographic content is prepared using Adobe software tools such as PhotoShop. It is stored in a special format (PSD) which maintains the different layers of the image. Different layers can include the background, text, images added from other sources etc. It is not unusual for a high end image for the publishing or web-publishing industries to include over 50 separate layers. The MCS supports the PSD format as an input format, thereby preserving the original quality and enabling the content based detection module to handle each layer separately .

~~IV.I.~~ *Message interface*

[635] The MPS supports two distinct interfaces to the MMSC/external applications:

[636] 1. An XML-RPC/HTTP interface, enabling platform and operating system decoupling between the MMSC and the MPS. This interface, documented in the attached ICD, enables complete control over the manipulation and conversion operations of the media objects and works at the media level.

[637] 2. A 3GPP standard, message-based interface designed in to make the integration of the UCnGO MPS as standard as possible for the OEM MMSC integrator or the VAS provider. The interface is based on the MM7/MM4/MM1 protocols. Using this interface, a complete unchanged MMS/Email message as received from the WAP GW/the other operator's MMSC/ the VAS can be sent as is to the MCS, and the response from the MCS can be sent as is to the recipient phone/MMSC/VAS server.

[638] Specifications:

[639] The SMTP protocol (default port 25, configurable) or the HTTP POST protocol (configurable port) is used to transfer the message for processing. The message can be any standard MM1/MM4/MM7 message is defined in the 3GPP TS23.140 Release 5.20. document. The target device(s) type identification can be performed in the following manners:

[640] · The message header contains a set of (one or more) "X-MMS-UserAgent" or "X-MMS-UAprof" or "X-MMS-Model" descriptors – in this case these descriptors are taken as representing the model types for the different intended recipients as they appear in the message in the TO: data field.

[641] The message header contains no extra information about the target devices, but an LDAP based user/device database has been configured to supply device parameters based on a user's MSISDN or email address. In this case the MCS performs an LDAP query for each target recipient specified in the "TO:" field of the message in order to find out the recipient terminal type.

[642]

[643] The processed response in returned in one of the following manners:

[644] The MCS can be configured to send the processed messages to a target SMTP server as MM7/MM4 messages. This way the MCS can sit between the external VAS/external MMSC and the local MMSC with no configuration changes.

[645] The MCS can be configured to send the processed messages via HTTP POST to a target server as an MM1 message. This way the MCS can sit between the WAP GW and the MMSC MMS proxy.

[646] The processed response(s) will be sent in MIME multipart format, with the presentation layer and media objects converted based on the recipient device. For example, for a WAP phone the presentation layer will be in the text-wml MIME type, and images will be in GIF/WBMP format. The message will be submitted once per each target device, since the content for the different target devices is now different, having undergone conversion. So for example an incoming MM7 message targeted at four recipients will generate four MM7 messages with one recipient each.

~~IV.J.~~—Presentation level conversion

[647] Effective multimedia presentation requires some information on the spatial and temporal relations between the different media objects presented. This functionality is performed by the presentation layer – HTML in web pages and Email, WML in WAP pages, SMIL in MMS messages. Some multimedia formats (e.g. EMS) and phones (e.g. Nokia 3510, Nokia 7210) do not support an explicit presentation language, but rather display the different media objects according to their own pre-defined logic.

[648] This means that in addition to the media objects conversion, the presentation level description has to be converted. In more complex cases (e.g. when no presentation description language exists) the actual media conversion has to be changed to account for the presentation logic. Some examples are:

[649] An image and accompanying text is to be sent to a WAP phone. By changing the image size target one can guarantee that the text will be able to be viewed on the screen together with the image without scrolling. This requires knowledge of the phone's effective (versus physical) display size, and control of the image size in pixels, the WML description (e.g. the align='left' directive for the text), etc. Thus, the generated WML deck should contain the proper parameters.

[650] An MMS message containing a cartoon with a set of images and associated text, alongside with SMIL description is sent to a Nokia 3510 which does not support SMIL. There is no specific order for the media

objects – that is, the last image (the cartoon's "punchline") can be the first image in the MMS message. Hence, if the SMIL description is just omitted, the received content will be worthless. The presentation level logic in this case has to order the media objects according to their desired display order as specified in the SMIL description, so that the Nokia user will get the cartoon in the proper sequence.

[651] Specifications

[652] The UCnGO MCS provides for the presentation level conversion for the SMIL,MIME multipart, HTML,WML, EMS formats (see Fig. 37 for the supported conversion matrix). Of these, SMIL and MIME multipart are supported as input formats, and all are supported as output formats.

[653]

[654] The supported conversion operations include:

[655] 1. Relative positioning of media objects based on target screen size.

[656] 2. Formatting of text (Bold, Italic, underlined, size) – in EMS, WML,HTML, MIME/HTML.

[657] 3. Picture and animation smart compress – that is, the images and animations are adapted to fit in pixel size and total file size based on the directives of

[658] 4. Audio smart compress for WAV/AMR/AAC, truncation for ringtones.

[659] 5. Video frame rate reduction, resizing.

[660] 6. Orientation change for images/video based on screen size.

[661] 7. Different scaling based on parameters – use viewable area, use scrollable area, target application type (WAP/MMS etc).

[662] Supported SMIL tags include: root-layout, region (and relevant tags), dur.

~~IV.K.1.~~ High Level Code for transcoding to a Picture Message

[663] %This is an algorithm in pseudocode for converting any input image into a B/W image with a maximum size of 72 by 28 pixels -

[664] %The maximum size for a Nokia Smart Messaging Picture Message

[665] function I4=newdither(fn,dorot),

[666] % if dorot is specified the image is rotated by 90 degrees to make it fit the screen aspect ratio better.

[667] I=imread(fn);I=double(I)/256;

[668] if exist('dorot'),I=imrotate(I,90,'nearest');end;

[669] if isgray(I),It=zeros(size(I,1),size(I,2),3);It(:,:,1)=I;It(:,:,2)=I;It(:,:,3)=I; I=It;end; %we make sure the image is 3 channel

[670] %RGB image even if it started out as a grayscale image.

[671] I0=rgb2hsv(I);It=I0;

[672] for i=1:3,I0(:,:,i)=filter2(ones(6,6)/36,I0(:,:,i),'same');end;

[673] It(:,:,3)=It(:,:,3)-0.75*I0(:,:,3);% We perform an unsharp operation in the luminance channel of the image

[674] mx=max(max(It(:,:,3)));

[675] mn=min(min(It(:,:,3)));

[676] It(:,:,3)=histeq(It(:,:,3));%We stretch the histogram of the luminance channel

[677] I=hsv2rgb(It);% And then convert back to RGB

[678] I=imresize(I,73/(size(I,2))*[0.8*size(I,1) size(I,2)],'bicbuic');

[679] I=I(:,2:end,:); %The image is resized, and eliminate the edge due to edge-scaling effects.

[680] mx=max(I(:));

[681] mn=min(I(:));

[682] I=(I-mn)/(mx-mn);

[683] I0=rgb2hsv(I);It=I0;

[684] for i=1:3,I0(:,:,i)=filter2(ones(6,6)/36,I0(:,:,i),'same');end;

[685] It(:,:,3)=It(:,:,3)-0.5*I0(:,:,3);

[686] mx=max(max(It(:,:,3)));

[687] mn=min(min(It(:,:,3)));

[688] It(:,:,3)=histeq(It(:,:,3));%(It(:,:,3)-mn)/(mx-mn);

[689] I=hsv2rgb(It);%We repeat the same transformation for the resizee image with different unsharp parameters.

[690] cy=floor(size(I,1)/2);

[691] I1=I(cy-min(13,cy-1):cy+min(14,cy),:,:);

[692] [I2,M]=rgb2ind(I1,4,'dither');% We reduce the image to 4 colors with dithering

[693] I2=(rgb2gray(ind2rgb(I2,M)));% Convert to grayscale

[694] I2=(I2-min(I2(:)))/(max(I2(:))-min(I2(:)));

[695] I3=zeros(size(I1,1),72,3);I3(:,:,1)=I2;I3(:,:,2)=I2;I3(:,:,3)=I2;

[696] I4=dither(I3,[0.25 0.25 0.25; 0.75 0.75 0.75;])>0; %finally we dither into 2 colors

[697] return

[698]

[699]

~~IV.K.2.~~ High Level Code for homogenized sharpening

[700]

[701] % smart_sharpen_homo -- Sharpening algorithm)

[702] %

[703] % Usage:

[704] % final_image=smart_sharpen_homo(initial_image,param)

[705] %

[706] % Inputs:

[707] % initial_image - RGB

[708] % param - parameters structure

[709] % param.Radius=[10,2]; %LpHomo Window

[710] % param.Power=[2,16]; %LpHomo Power

[711] % param.Weights=[1,1]; %LpHomo Weight

[712] % param.thr_gain=20; %- threshold-like gain for sharpening

[713] % param.thr_power=.3;

[714] % param.thr_window=5;

[715] %

[716] % Outputs:

[717] % final_image - RGB

[718] %

[719] %
[720] % See Also
[721] %
[722] % Written by Goldentayer Lev, March 2001
[723]
[724] for ch=1:size(initial_image,3)
[725] mpic=initial_image(:,:,ch);
[726] matVal=param.Radius(1);matrixDims = int32([matVal ,matVal ,matVal ,matVal]);
[727] J1=double(iplmex('ucngoLPHomogenizationFP',single(mpic),matrixDims,single(param.Power(1))));
[728] matVal=param.Radius(2);matrixDims = int32([matVal ,matVal ,matVal ,matVal]);
[729] J2=double(iplmex('ucngoLPHomogenizationFP',single(mpic),matrixDims,single(param.Power(2))));
[730] a0=param.Weights(1);a1=param.Weights(2);
[731] I=(mpic+a0*J1+a1*J2)/(a0+a1+1);
[732] I=(I-min(I(:)))/(max(I(:))-min(I(:)));
[733] K=ones(param.thr_window)/param.thr_window/param.thr_window;
[734] thr_mult=(max(min(abs(mpic-filter2(K,mpic))*param.thr_gain,1),0)).^param.thr_power;
[735] final_image(:,:,ch)=mpic.*(1-thr_mult)+I.*thr_mult;
[736] end
[737]

IV.K.3. High level code for image color palette and size adaptation

[739]
[740] % This is a pseudocode implementation of a smart compression algorithm - namely,
[741] % an algorithm that gets as its input an image file and outputs a version of the image in a specific format (GIF in this example)
[742] % and with a maximum file size that does not exceed the limitations dictated by the end device.
[743] % This implementation iterates the number of colors in the quantization step until the file is small enough.
[744] function err=smartcompress(filename),
[745] NumCol=[2 8 10 13 16 20 24 32]; %These are the numbers of colors in the colormap - can range from 2-256 for a typical 256 color
[746] %display. These quantized color number steps are dependent on the target device - one would choose differnet steps for other devices
[747] % in order to minimize the run time.
[748] SizeThreshold=2720; % This is the upper threshold in bytes on the output file size for a T68 using a WAP browser.
[749] Impath='D:\ImageMagick\ImageMagick-win2k\';
[750] netpbmpath='D:\netpbm\bin\';
[751] NumQuant=length(NumCol);

```
[752] I=imread(filename);I=double(I)/255; %we read the original file and
   convert it to a floating point image.
[753] Q=min(96/size(I,2),100/size(I,1));
[754] I1=zeros(size(I));
[755] for i=1:3,I1(:,:,i)=filter2(ones(3,3),I(:,:,i),'same');end; %We create a
   blurred (rectangular kernel) version of the original image.
[756] I=(double(I)-0.25/9*double(I1));I=I-I.*(I<0); %And then substract it
   from the original to create an unsharp operation.
[757] It=rgb2ycbcr(I);It=histeq(It);It=imadjust(It,[0 1],[0.1 0.9]);
   %conversion to luminance/color space to stretch the luminance histogram
[758] It(:,:,1)=histeq(It(:,:,1));It(:,:,2)=imadjust(histeq(It(:,:,2)),[0
   1],[0.2 0.8]);
[759] It(:,:,3)=imadjust(histeq(It(:,:,3)),[0 1],[0.2 0.8]);
[760] I=(ycbcr2rgb(It)); % we convert back to RGB colorspace after
   stretching the luminance channel.
[761] I=histeq(I);I=I.^0.75; %And then we further manipulate the
   histogram.
[762] I=imresize(I,Q,'bicubic'); %We resize to the target size in pixels (this
   could also be an iteration based on target file size).
[763] imwrite(I,'inp.jpg','Quality',100);
[764] dos([Impath 'convert -compress LZW ' pwd '\inp.jpg ' pwd '\inp.gif']);
[765] dos([netpbmpath 'giftopnm inp.gif > inp.ppm']); %We convert the
   resized image to a ppm file so we can run the ppmquant algorithm
[766] FileSize=zeros(1,NumQuant);
[767] MaxNum=2;
[768] for i=1:NumQuant,
[769]     dos([netpbmpath 'ppmquant ' num2str(NumCol(i)) ' inp.ppm >
   out.ppm']);
[770]     dos([netpbmpath 'ppmtogif out.ppm > out.gif']);
[771]     D=dir(['out.gif']);
[772]     FileSize(i)=D.bytes; %We measure the file size of the output image.
[773]     if (FileSize(i)<SizeThreshold),MaxNum=NumCol(i);end; % If it is
   valid (less than threshold) we update the maximum
[774]     %color number we could achieve
[775] end;
[776]     disp(MaxNum);
[777]     dos([netpbmpath 'ppmquant ' num2str(MaxNum) ' inp.ppm >
   out.ppm']);
[778]     dos([netpbmpath 'ppmtogif out.ppm > shid1.gif']);
[779]     dos('ftp -s:transfer.bat'); %we put the file in a directory with pre-
   source information (in this case a WML card deck) so that
[780]     %we can view the image on a target device
[781] end;
[782]
[783]
```

~~IV.K.4.~~ High Level Code for color adaptation to special color handsets (3-3-2)

[784]

```
[785] function final_image=v332_dither(initial_image)
[786] % v332_dither -- jphone dithering algorithm (for faces,cars etc)
[787] % This algorithm adapts to displays with a 3-3-2 color palette (R,G,B).
   This is
[788] % prevalent in Japanese color handsets
[789] %  Usage:
[790] %     final_image=v332_dither(initial_image)
[791] %
[792] %  Inputs:
[793] %   initial_image - RGB
[794] %
[795] %  Outputs:
[796] %    final_image - 332 image
[797] %
[798] %
[799] %  See Also
[800] %
[801] % Written by Goldentayer Lev, March 2001
[802]
[803] for ch=1:3
[804]
[805] initial_image_gray = initial_image(:,:,ch) ;
[806]
[807] %mpic=im2double(imresize(initial_image_gray,[92 98],'bicubic'));
[808] %mpic=im2double(imresize(initial_image_gray,[74 98],'bicubic'));
[809] resize_ratio=96/size(initial_image,2);
   mpic=im2double(imresize(initial_image_gray,resize_ratio,'bicubic'));
[810]
[811] mpic=unpair(mpic,[1 1 1 1]);
[812] mpic=(mpic+histeq(mpic))/2;
[813]
[814] matVal=2;matrixDims = int32([matVal ,matVal ,matVal ,matVal ]);
[815]
      J4=double(iplmex('ucngoLPHomogenizationFP',single(mpic),matrixDim
   s,single(16)  ));
[816]
[817] a=.3;mpic=(mpic+a*J4)/(1+a);
[818] rgb_image(:,:,ch)=mpic;
[819] end
[820]
[821]
[822] jj=0;
[823] for i1=1:8
[824]    for i2=1:8
[825]       for i3=1:4
[826]          jj=jj+1;
[827]          YMAP(jj,1)=(i1-1)*(1/8)+(1/16);
```

```
[828]         YMAP(jj,2)=(i2-1)*(1/8)+(1/16);
[829]         YMAP(jj,3)=(i3-1)*((1-1/8)/3)+(1/16);
[830]      end
[831]   end
[832] end
[833]
[834]
```

IV.K.5 High Level Code for JPEG artifacts removal

```
[835]
[836] function final_image=jpeg_filter(initial_image)
[837] % jpeg_filter -- Jpeg artifacts filter
[838] %
[839] %  Usage:
[840] %    final_image=jpeg_filter(initial_image)
[841] %
[842] %  Inputs:
[843] %   initial_image - RGB
[844] %
[845] %  Outputs:
[846] %   final_image - RGB
[847] %
[848] %
[849] %
[850] %  See Also
[851] %
[852] % Written by Goldentayer Lev, March 2001
[853]
[854] xpic=im2double(initial_image);
[855] for ch=1:size(xpic,3)
[856]    xfilt(:,:,ch)=filter2(ones(2)/4,xpic(:,:,ch));
[857] end
[858] for ch=1:size(xpic,3)
[859]    J0=xfilt(:,:,ch);
[860]    matVal=6;matrixDims = int32([matVal ,matVal ,matVal ,matVal ]);
[861]
   J1=double(iplmex('ucngoLPHomogenizationFP',single(xfilt(:,:,ch)),matrixD
   ims,single(8)  ));
[862]    matVal=2;matrixDims = int32([matVal ,matVal ,matVal ,matVal ]);
[863]
      J2=double(iplmex('ucngoLPHomogenizationFP',single(xfilt(:,:,ch)),matr
   ixDims,single(16)  ));
[864]    a0=2;a1=.5;a2=.1;
[865]    xenh(:,:,ch)=(a0*J0+a1*J1+a2*J2)/(a0+a1+a2);
[866] end
[867] for ch=1:size(xpic,3)
[868]    final_image(:,:,ch)=wiener2(xenh(:,:,ch),[3 3]);
[869] end
```

```
[870]
[871] final_image=(max(min(final_image,1),0).*(xpic+.25)+xpic.*(1-
   xpic))/1.25;
[872]
```

IV.K.6. High Level Code for image content detection

```
[874]
[875] function content_shell(varargin)
[876] if(nargin==0)
[877]     figure
[878]     set(gcf,'Name','Image Content Detection Shell');
[879]     set(gcf,'MenuBar','none');
[880]
   HM_util=uimenu(gcf,'Label','Execute','Callback','content_detect_v4(''exec
   ute'')');
[881] else
[882]     [in_file, in_path] = uigetfile('*.*');
[883]     pause(.2);
[884]     try
[885]         set(gcf,'Pointer','watch');
[886]         content_type=content_detect(in_file, in_path);
[887]         set(gcf,'Pointer','arrow');
[888]         %if(0)
[889]     catch
[890]         beep;
[891]         disp(lasterr);
[892]         %    disp('Invalid file name');
[893]         set(gcf,'Pointer','arrow');
[894]     end
[895] end
[896] return
[897]
[898]
[899] function content_type=content_detect(in_file, in_path)
[900] image_name=[in_path,in_file ];
[901] in_img=content_read(image_name);
[902] col_sat=content_color_sat(in_img);
[903] background_vect=content_background(in_img);
[904] dither_measure=content_dither(in_img);
[905] unique_colors=content_colors(in_img);
[906]
[907] if(unique_colors==2)
[908]     grad_vect=content_grad_stat(filter2(ones(5,5)/25,in_img(:,:,1)));
[909] end
[910] if(grad_vect(2)>grad_vect(1)*1)
[911]     res='Photographical Image;';
[912] else
[913]     res='Synthetic Image; ';
```

```
[914]    if(grad_vect(1)>grad_vect(3)*1)
[915]
[916]       if(dither_measure>.01)
[917]          res=[res, 'Texture; '];
[918]       end
[919]    end
[920]    if(col_sat>0.04)
[921]       res=[res, 'Saturated Colors; '];
[922]    end
[923] end
[924]
[925]
[926] if(unique_colors<=2)
[927]    res=[res, 'Binary; '];
[928] elseif(mean2(max(in_img,3)-min(in_img,3))==0)
[929]    if(unique_colors<=256)
[930]       res=[res, num2str(unique_colors), ' Colors; '];
[931]    else
[932]       res=[res, 'Rich Color; '];
[933]    end
[934] else
[935]    res=[res, 'Grayscale; '];
[936] end
[937]
[938] if(background_vect(1)>0.25)
[939]    res=[res, 'Background Color Detected; '];
[940] end
[941]
[942]
[943] disp(res);
[944]
[945]
[946] %grad_vect
[947] stat=['Edge: ',num2str(grad_vect(1)),' \newline ',...
[948]       'Curved: ',num2str(grad_vect(2)),' \newline ',...
[949]       'Smooth: ',num2str(grad_vect(3)),' \newline ',...
[950]       'Texture: ',num2str(dither_measure),' \newline ',...
[951]       'Unique Colors: ',num2str(unique_colors),' \newline ',...
[952]       'Color Saturation: ',num2str(col_sat),' \newline ',...
[953]       'Background Area: ',num2str(background_vect(1)),' \newline ',...
[954]       'Background Color: ',num2str(background_vect(2:end))];
[955]
[956]
[957] %figure(1),
[958] %clf;
[959] subplot(1,2,1);
[960] imshow(in_img);
```

```
[961] title([in_file]);
[962] subplot(1,2,2);
[963] set(gcf,'Units','normalized')
[964] imshow(ones(128,256)*0.9);
[965] title(['\it',res]);
[966] hh=text(20,70,stat);
[967] set(hh,'FontSize',8);
[968]
[969] %disp('kaki')
[970] content_type=res;
[971] return
[972]
[973]
[974] function    in_img=content_read (image_name)
[975] if(isempty(image_name)),
[976]     image_name='dolyparton.jpg';
[977] end
[978]
[979] aa=imfinfo(image_name);
[980] if(strcmp(aa.ColorType,'truecolor'))
[981]     in_img=im2double(imread(image_name));
[982] else
[983]     [A,Map]=imread(image_name);
[984]     in_img=im2double(ind2rgb(A,Map));
[985]     clear A; clear Map;
[986] end
[987] disp(['Image: ',image_name]);
[988] return
[989]
[990]
[991] function col_sat=content_color_sat(in_img)
[992] dv_img=std(in_img,1,3);
[993] ssi=prod( size(in_img));
[994] %disp([         ' color saturation: ',num2str(sum(dv_img(:))/ssi)]);
[995] col_sat=sum(dv_img(:))/ssi;
[996] return
[997]
[998] function background_vect=content_background(in_img)
[999] nchannels=size(in_img,3);
[1000]        img_area=size(in_img,1)*size(in_img,2);
[1001]        [A,M]=rgb2ind(in_img,256);
[1002]        [aa,ind]=max(imhist(A,M));
[1003]        background_vect(1)=aa/prod(size(A));
[1004]        background_vect(2:1+nchannels)=M(ind);
[1005]        disp([         ' Background size: ',aa/prod(size(A))]);
[1006]        disp([         ' Background color: ',num2str(M(ind))]);
[1007]        return
```

```
[1008]
[1009]
[1010]    function dither_measure=content_dither(in_img)
[1011]    nchannels=size(in_img,3);
[1012]    for(ch =1:nchannels)
[1013]        tmp_img=in_img(:,:,ch);
[1014]        dither_score(ch)=mean2(abs(tmp_img-
  medfilt2(tmp_img,[3,3])));
[1015]    end
[1016]    dither_measure=mean(dither_score);
[1017]    disp(['Dither score', num2str(dither_score)]);
[1018]    return
[1019]
[1020]    function unique_colors=content_colors(in_img)
[1021]    if(sum(in_img(:)-(in_img(:)>0)))
[1022]        [A,M]= cmunique(in_img);
[1023]        unique_colors=length(M);
[1024]    else
[1025]        if(sum((in_img(:)-in_img(1)).^2))
[1026]            unique_colors=2;
[1027]        else
[1028]            unique_colors=1;
[1029]        end
[1030]    end
[1031]    return
[1032]
[1033]
[1034]    function grad_vect=content_grad_stat(in_img)
[1035]    nchannels=size(in_img,3);
[1036]
[1037]
[1038]    nhood=ones(5,5);
[1039]    per_up=floor(sum(nhood(:))*.75);
[1040]    per_dn=ceil(sum(nhood(:))*.25);
[1041]    if(per_up<=per_dn)
[1042]        per_up=sum(nhhod(:));
[1043]        per_dn=1;
[1044]    end
[1045]
[1046]    the_img=in_img;
[1047]
[1048]    for(ch=1:nchannels),
[1049]        grad(:,:,ch)=(ordfilt2(the_img(:,:,ch),per_up,nhood)-
  ordfilt2(the_img(:,:,ch),per_dn,nhood));
[1050]    end
[1051]
[1052]    thr_edge=.085;
```

[1053] thr_noise=.02;
[1054] edge_grad=(grad> thr_edge);
[1055] shape_grad=((grad<= thr_edge)&(grad>= thr_noise));
[1056] noise_grad=((grad< thr_noise)&(grad>= thr_noise));
[1057]
[1058] for(ch=1:nchannels),
[1059] mask_grad(:,:,ch)=(shape_grad(:,:,ch) & (filter2(nhood,edge_grad(:,:,ch))==0));
[1060] end
[1061]
[1062] ssi=prod(size(in_img));
[1063]
[1064]
[1065] grad_vect(1)=sum(edge_grad(:))/ssi;
[1066] grad_vect(2)=sum(shape_grad(:))/ssi;
[1067] grad_vect(3)=sum(noise_grad(:))/ssi;
[1068] grad_vect(4)=sum(mask_grad(:))/ssi;
[1069] return

~~IV.L.~~ *Caching and Smart Forwarding*

[1070] Certain distribution circumstances in MMS lead to situation where the same content is sent to a large number of devices with similar or identical characteristics. Examples are:

[1071] A VAS (e.g. a stock quote provider) sends an update to thousands of subscribers at the same time – in this situation hundreds of them will have identical phones and therefore media conversion should not be repeated for each one.

[1072] Superdistribution – a user gets a funny message and forwards it to his buddies, who then send it onward to their friends etc. Since some of the recipients may use the same device, there is no need to keep converting the message again and again. Furthermore, the degradations caused by continuous conversion should and can be avoided. See for example figure 32– the 2nd recipient has a color enabled display, and hence should get a color image/video. This can only happen if smart forwarding is implemented.

[1073] In short, caching means that when a new transcoding/conversion request arrives, the MPS looks in the cache to see if an identical/practically identical request for transcoding of the same media object into an identical/practically identical device has been submitted in the past and if the result of this operation is still in the cache. If so, the URI of the object in the cache is returned as the transcoded result and the actual transcoding operation is avoided.

[1074]

[1075] Smart forwarding is similar to regular caching, except that the MPS retrieves the cached transcoded result based on the original media object. That is, if a content request for object "B" (originally transcoded from object "A") to device "T" is requested, the MPS retrieves the cached result of "A" transcoded into "T", not of "B" transcoded into "T".

[1076] Other modifications and variations to the invention will be apparent to those skilled in the art from the foregoing disclosure and teachings. Thus, while only certain embodiments of the invention have been specifically described herein, it will be apparent that numerous modifications may be made thereto without departing from the spirit and scope of the invention.

WHAT IS CLAIMED IS

1. An MMS communication system for displaying images on a display terminal of a mobile or portable communication device, the system comprising:
an input adapted to receive pre-source information;
a transmitter adapted to transmit the pre-source information;
a server adapted to receive the transmitted pre-source information and further adapted to convert the pre-source information to source information suitable for display on the display terminal; and
a source transmitter adapted to transmit the source information to the display terminal.

2. The system of claim 1, wherein the server further comprises a display characteristics identifier adapted to identify display characteristics of the display terminal.

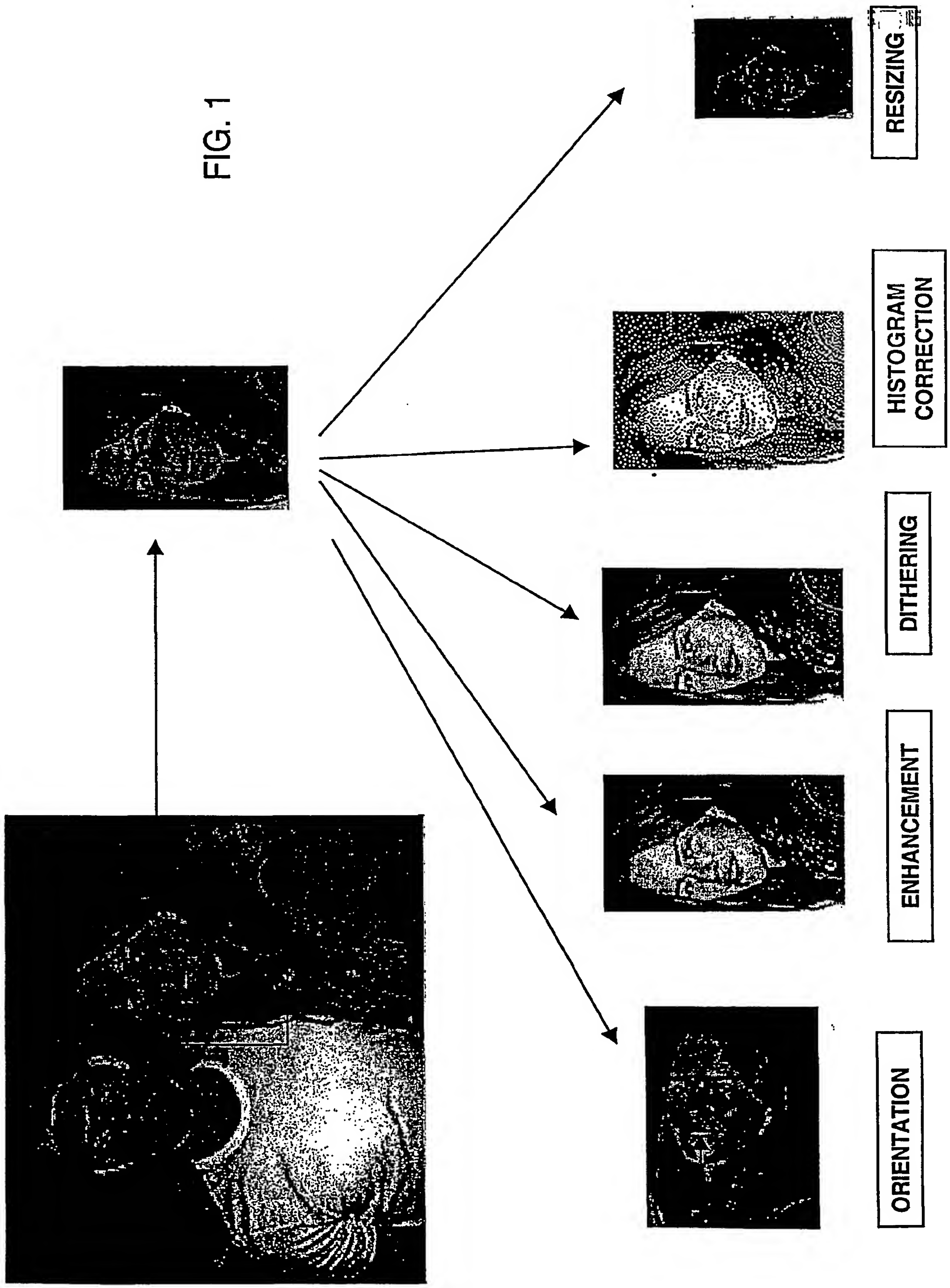
3. The system of claim 1, wherein the server further comprises a pre-source transcoder adapted to transcode pre-source information to source information.

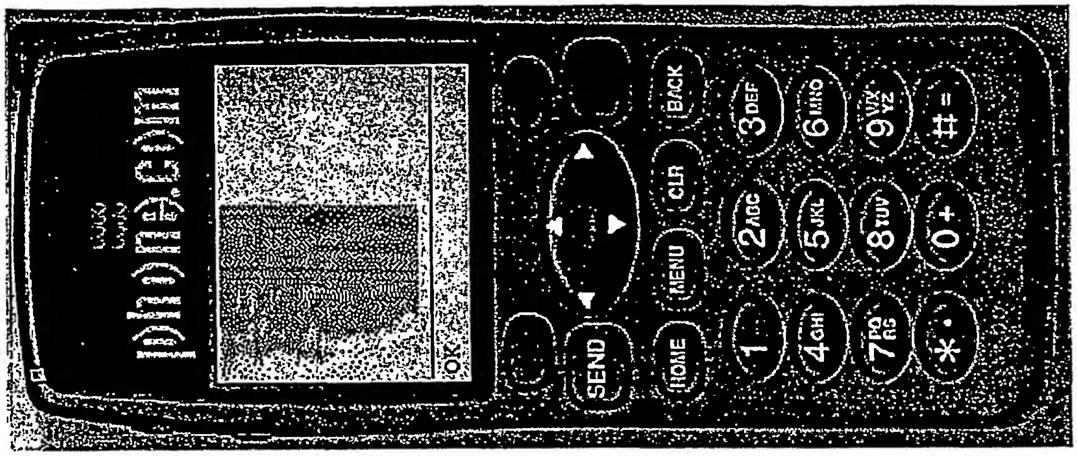
4. The system of claim 1, wherein the served comprises at least one source transcoder adapted to transcode a first source information in a first format to a second source information in a second format.

5. The system of claim 4, wherein the second format corresponds to a display format for the display terminal.

6. A method for enabling communication of MMS information suitable for display on a mobile or portable communication terminal, comprising:
inputting pre-source information;
transcoding pre-source information into source information;
transcoding source information another source format suitable for display on the target display terminal;
transmitting the transcoded source information to the target display terminal;
displaying the transmitted and transcoded source information on the display space of a target display terminal in a format most suitable for that particular display terminal.

FIG. 1


RESIZING
HISTOGRAM CORRECTION
DITHERING
ENHANCEMENT
ORIENTATION



PANORAMA

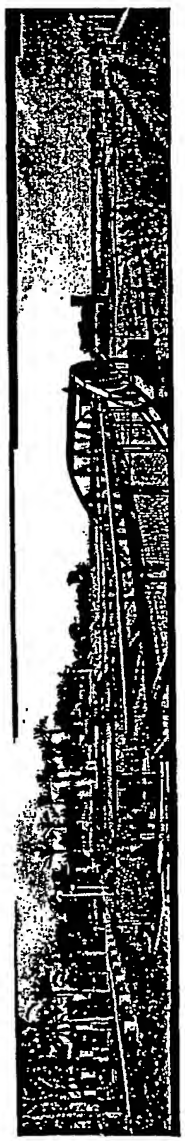


FIG. 2

Content Based Conversion

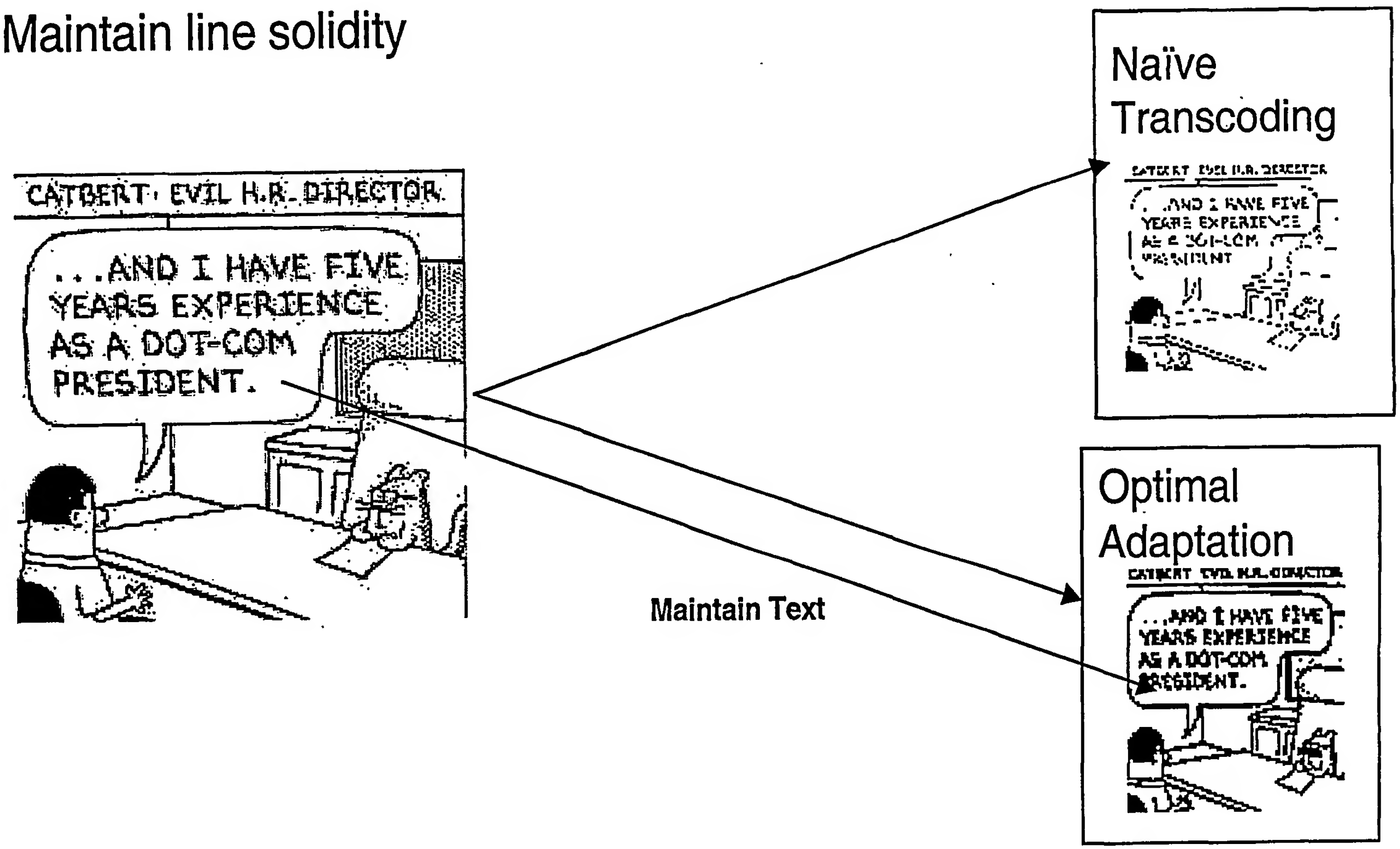
Cartoons:

Different processing for text and drawing

Ignore gray level information

Maintain line solidity

FIG. 3



Content Based Conversion

Stock Charts:

Remove superfluous information

Maintain line solidity

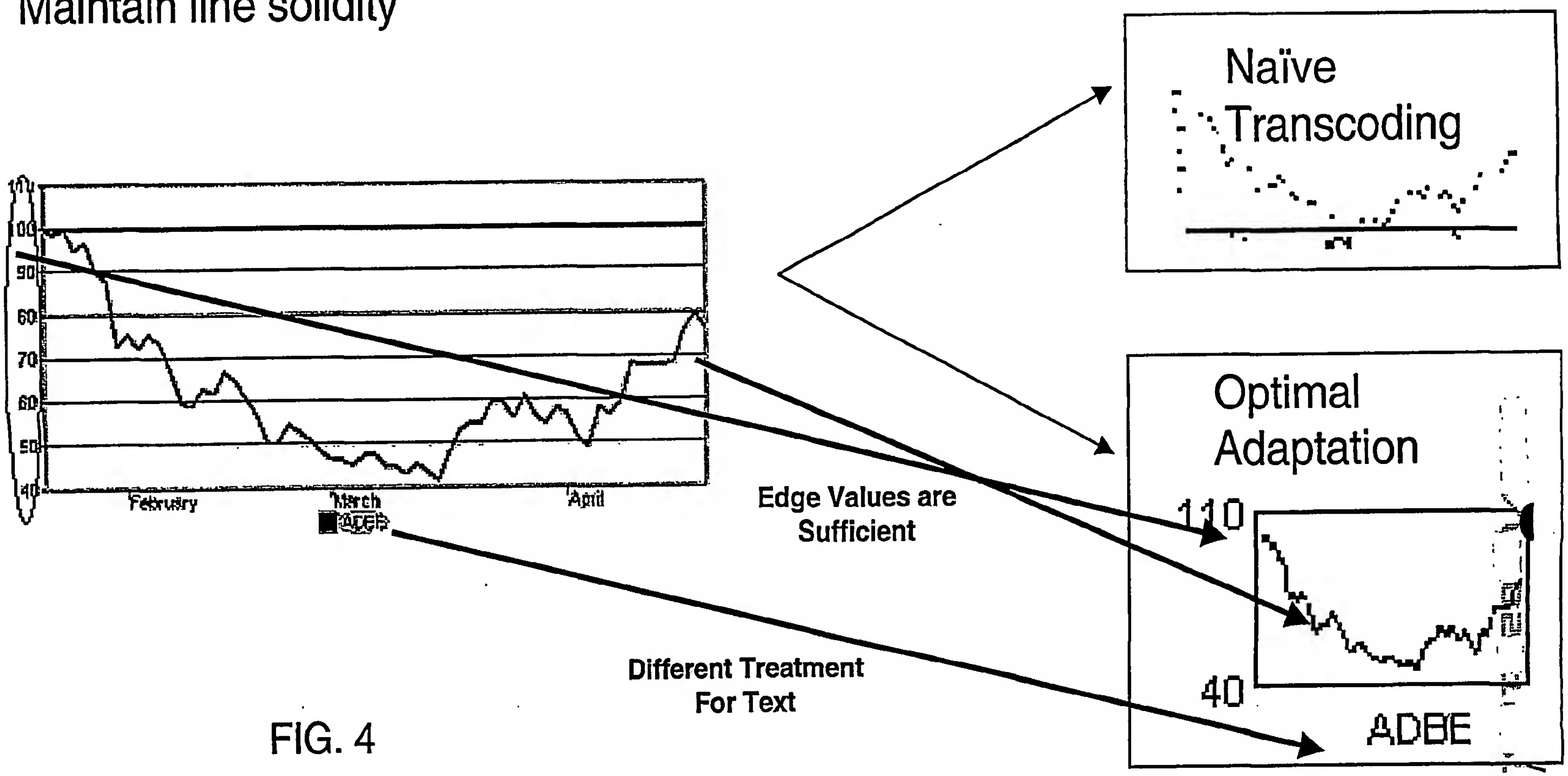


FIG. 4

UCnGO - Image Messaging Server

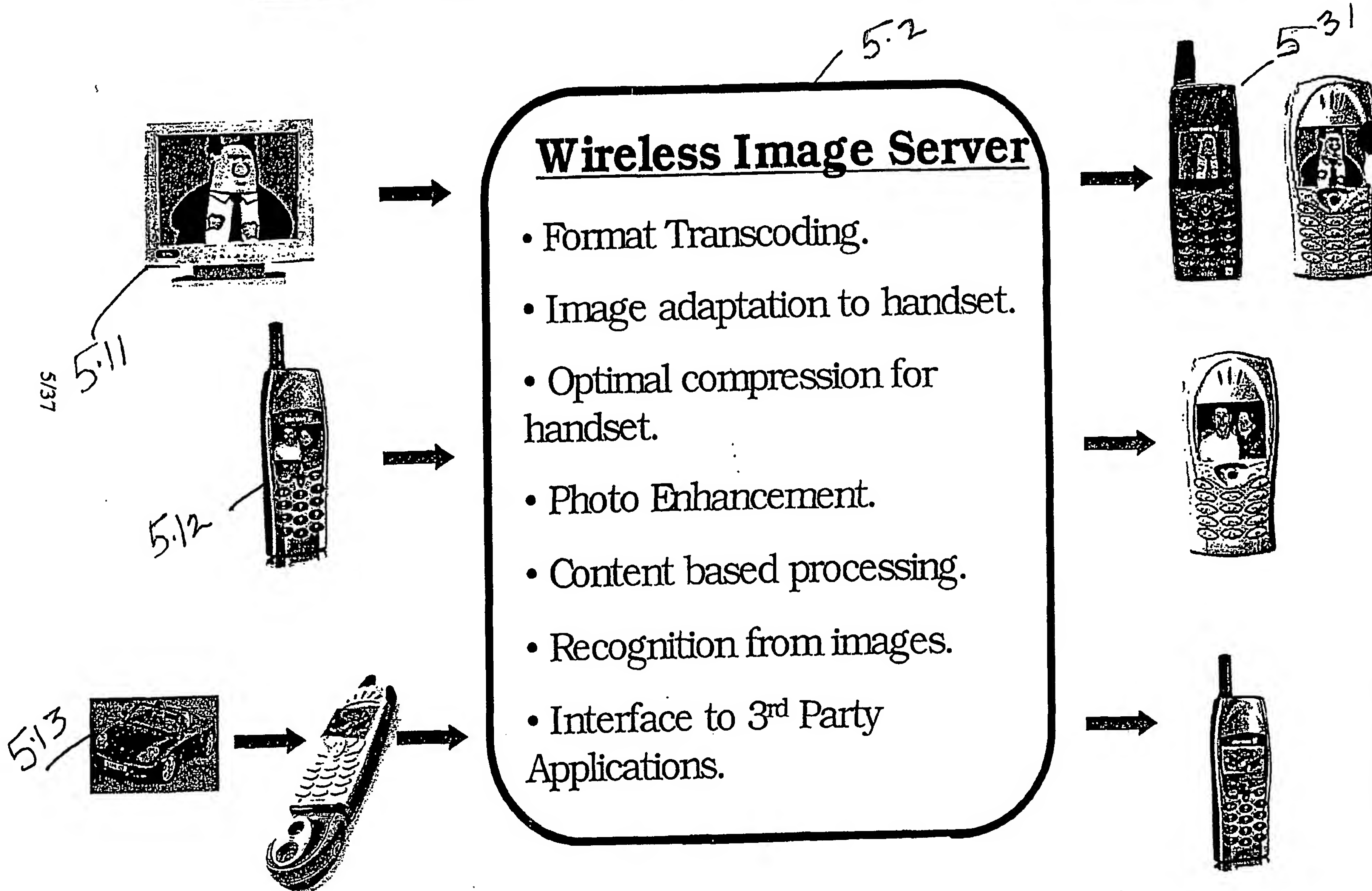


FIG 5

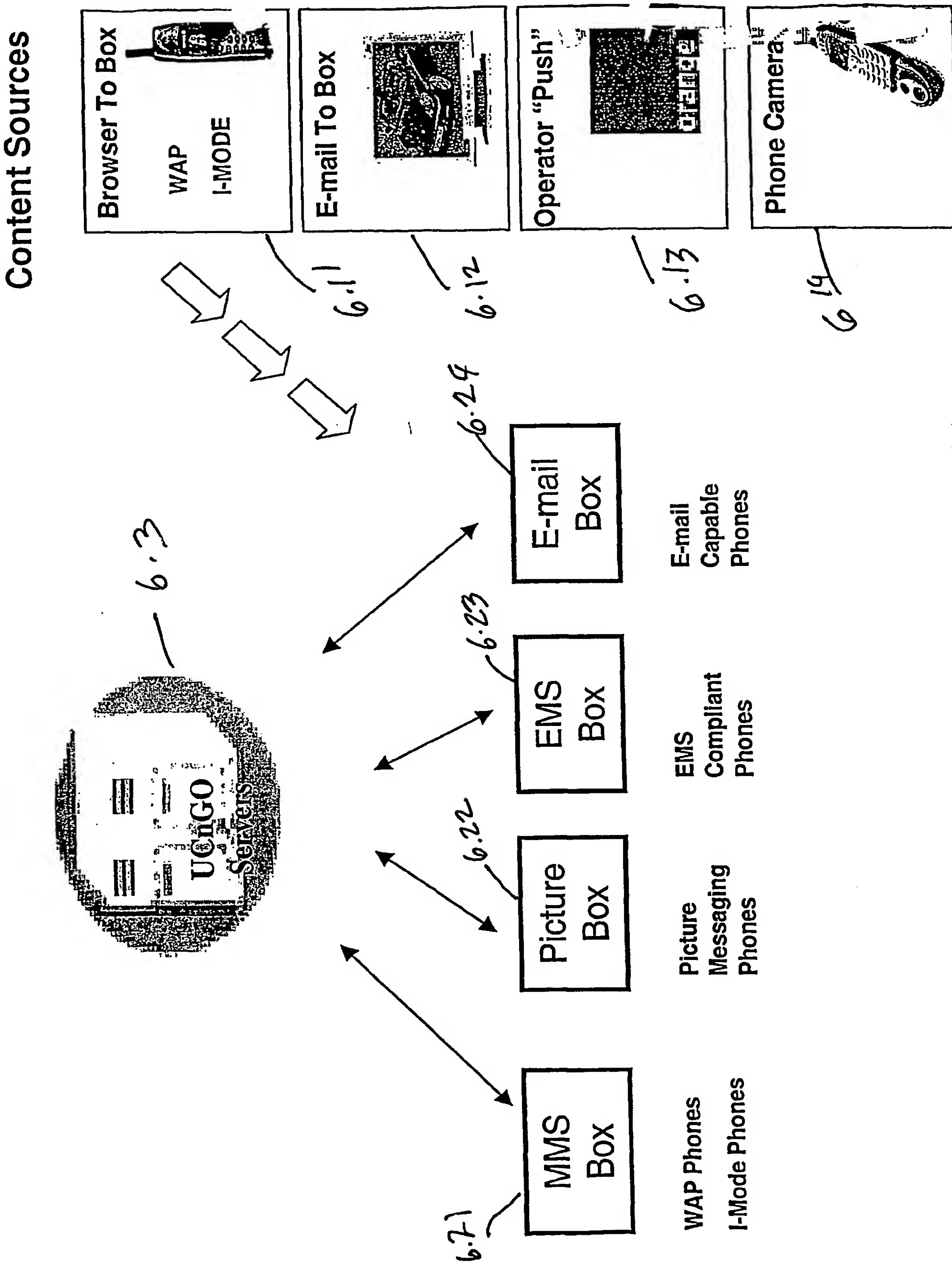

Content Sources
Browser To Box
WAP
I-MODE
E-mail To Box
Operator "Push"
Phone Camera
6.11
6.12
6.13
6.14
UCnGO
Servers
6.3
MMS Box
Picture Box
EMS Box
E-mail Box
6.21
6.22
6.23
6.24
WAP Phones
I-Mode Phones
Picture Messaging Phones
EMS Compliant Phones
E-mail Capable Phones

FIG. 6

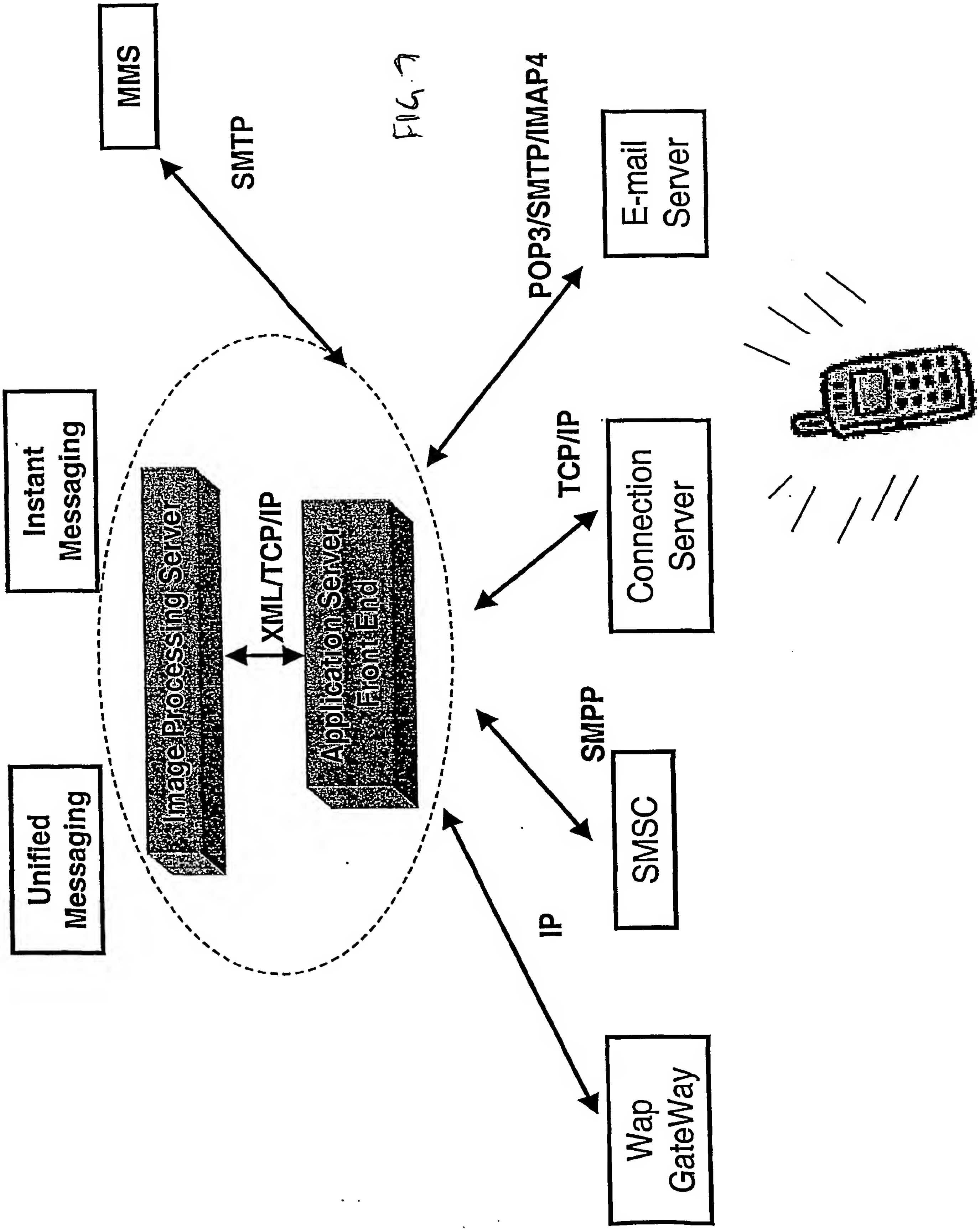

MMS
SMTP
Instant Messaging
Unified Messaging
Image Processing Server
XML/TCP/IP
Application Server Front End
POP3/SMTP/IMAP4
E-mail Server
TCP/IP
Connection Server
SMPP
SMSC
IP
Wap GateWay

FIG. 7

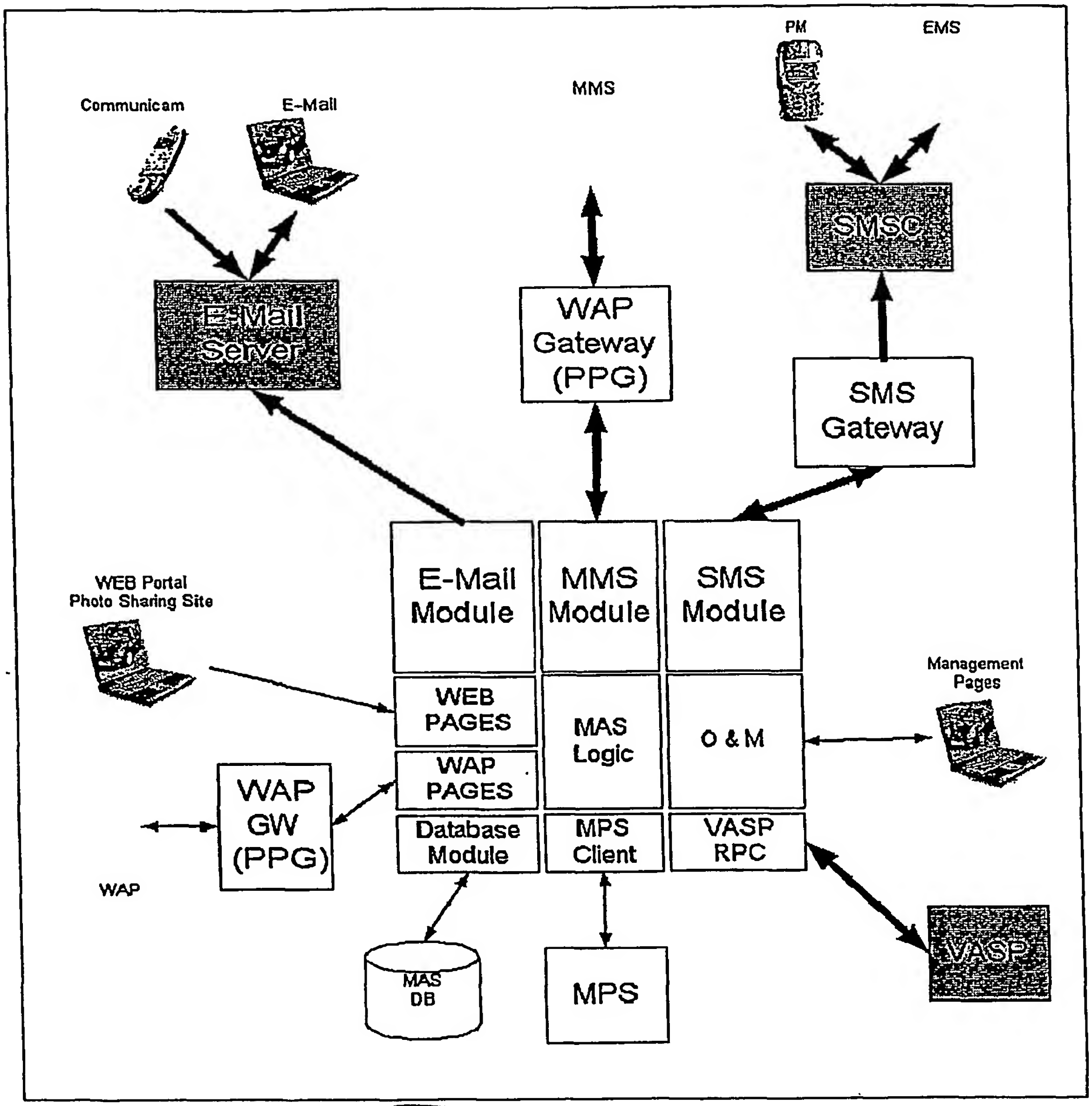

Communicam
E-Mail
MMS
PM
EMS
E-Mail Server
WAP Gateway (PPG)
SMSC
SMS Gateway
E-Mail Module
MMS Module
SMS Module
WEB Portal Photo Sharing Site
WEB PAGES
WAP PAGES
MAS Logic
O & M
Management Pages
WAP GW (PPG)
Database Module
MPS Client
VASP RPC
WAP
MAS DB
MPS
VASP

FIG. 8

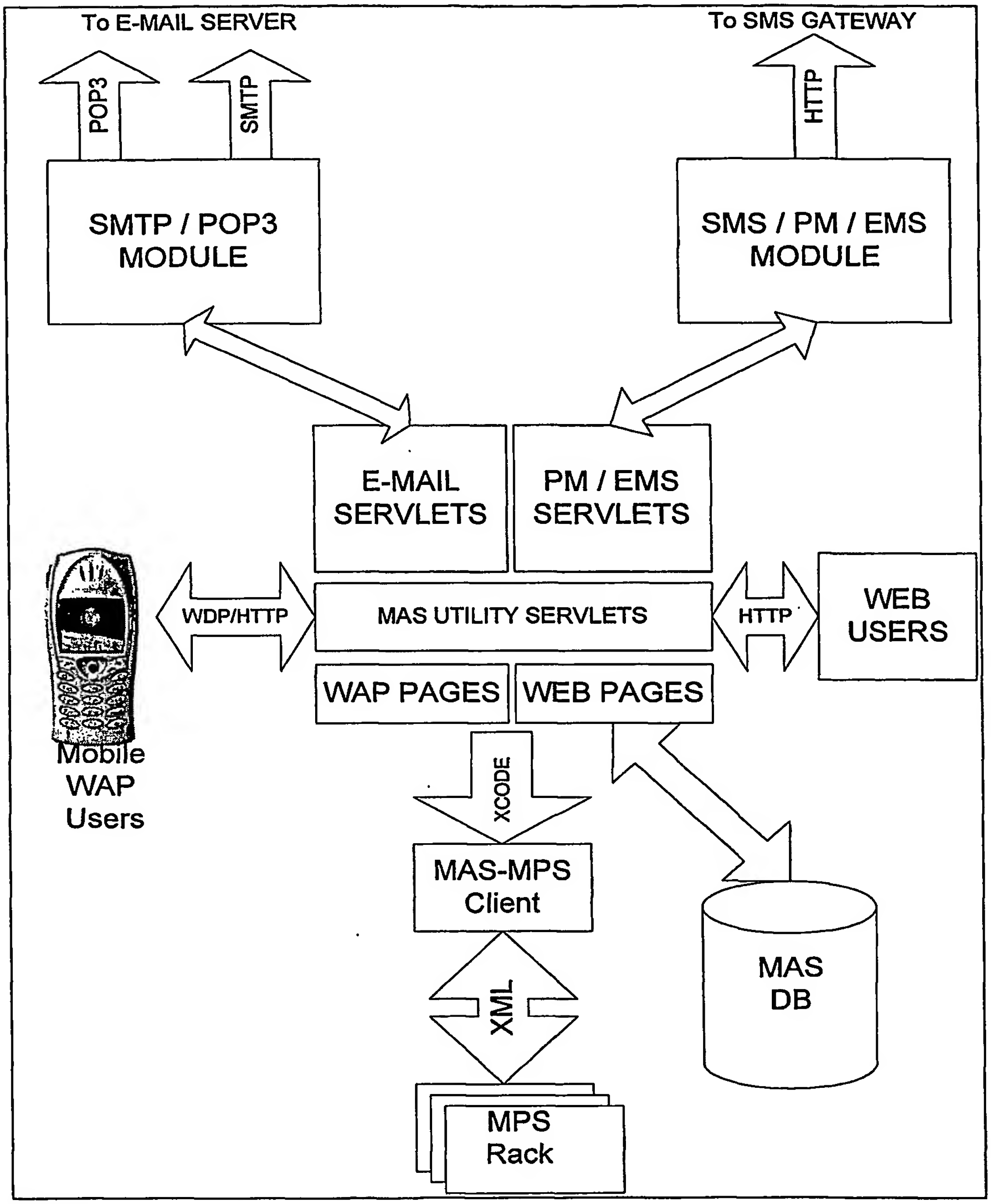

To E-MAIL SERVER
POP3
SMTP
To SMS GATEWAY
HTTP
SMTP / POP3 MODULE
SMS / PM / EMS MODULE
E-MAIL SERVLETS
PM / EMS SERVLETS
WDP/HTTP
MAS UTILITY SERVLETS
HTTP
WEB USERS
WAP PAGES
WEB PAGES
Mobile WAP Users
XCODE
MAS-MPS Client
XML
MAS DB
MPS Rack

FIG.9

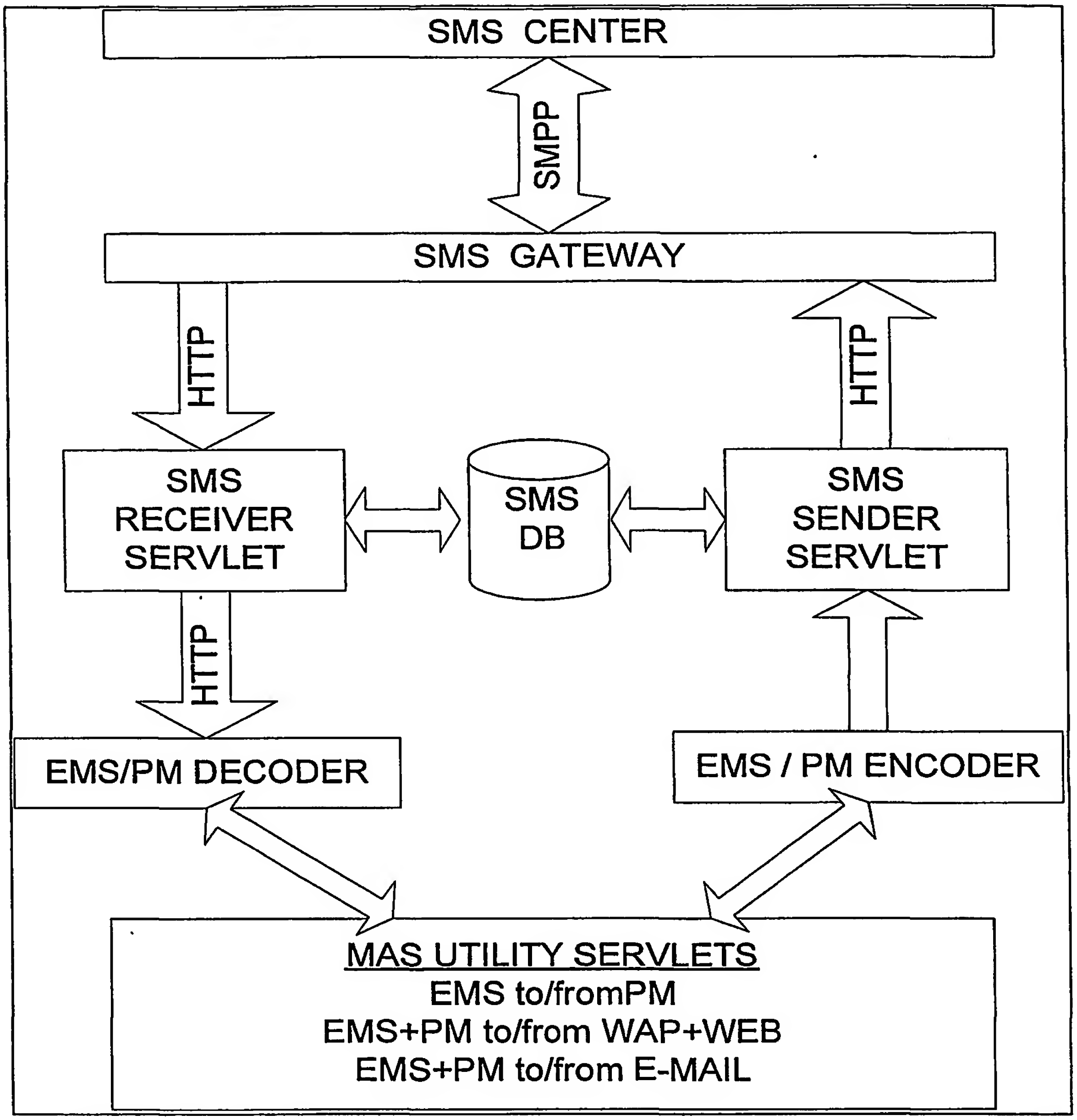

SMS CENTER
SMPP
SMS GATEWAY
HTTP
HTTP
SMS RECEIVER SERVLET
SMS DB
SMS SENDER SERVLET
HTTP
EMS/PM DECODER
EMS / PM ENCODER
MAS UTILITY SERVLETS
EMS to/fromPM
EMS+PM to/from WAP+WEB
EMS+PM to/from E-MAIL

FIG.10

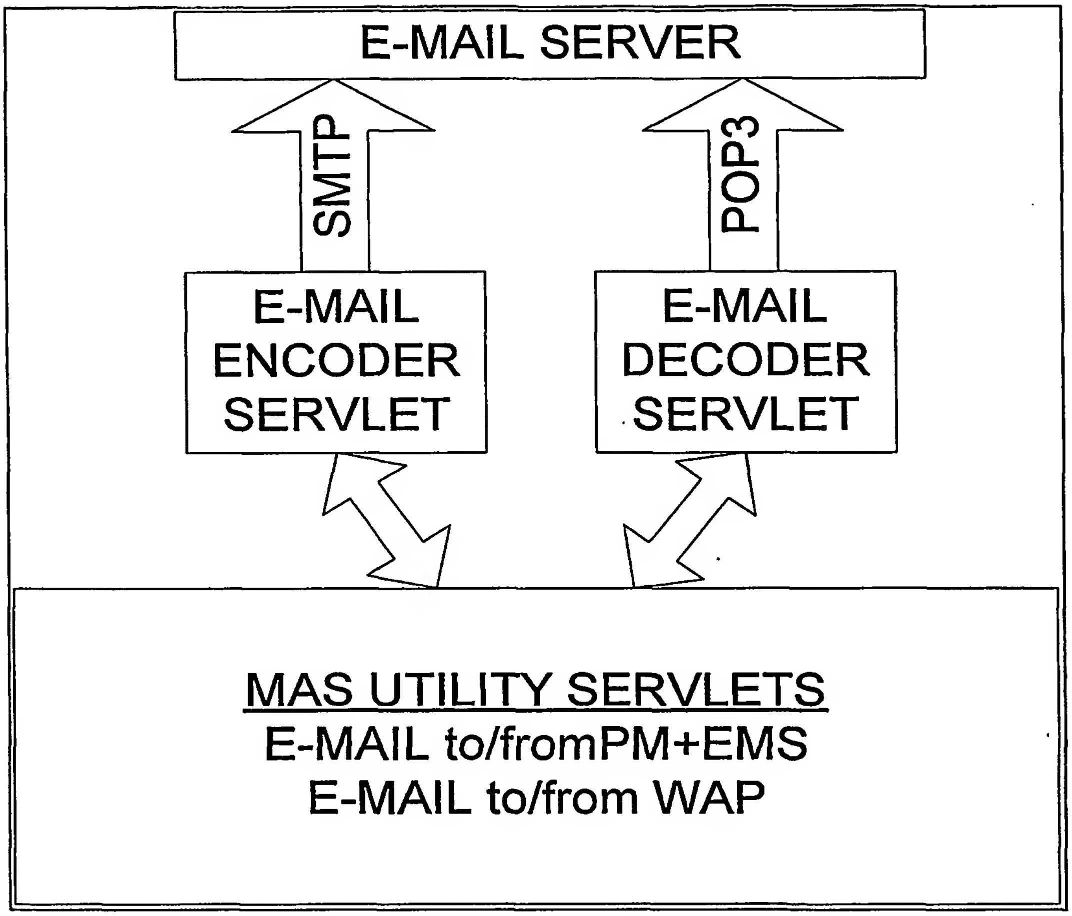

E-MAIL SERVER
SMTP
POP3
E-MAIL ENCODER SERVLET
E-MAIL DECODER SERVLET
MAS UTILITY SERVLETS
E-MAIL to/fromPM+EMS
E-MAIL to/from WAP

FIG.11

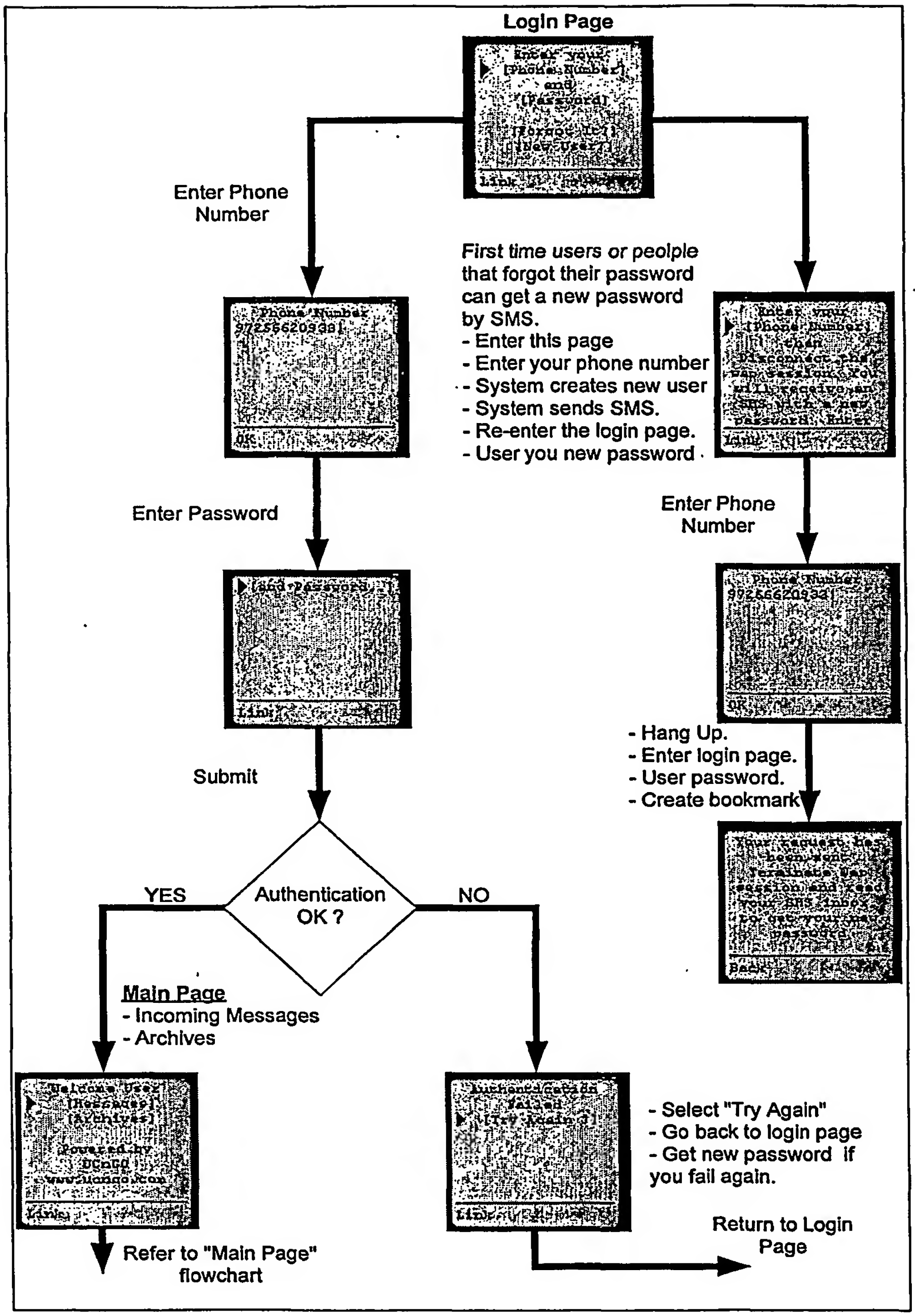

Login Page
Enter Phone Number
First time users or peolple that forgot their password can get a new password by SMS.
- Enter this page
- Enter your phone number
- System creates new user
- System sends SMS.
- Re-enter the login page.
- User you new password
Enter Password
Enter Phone Number
Submit
- Hang Up.
- Enter login page.
- User password.
- Create bookmark
YES
Authentication OK ?
NO
Main Page
- Incoming Messages
- Archives
- Select "Try Again"
- Go back to login page
- Get new password if you fail again.
Return to Login Page
Refer to "Main Page" flowchart

FIG.12

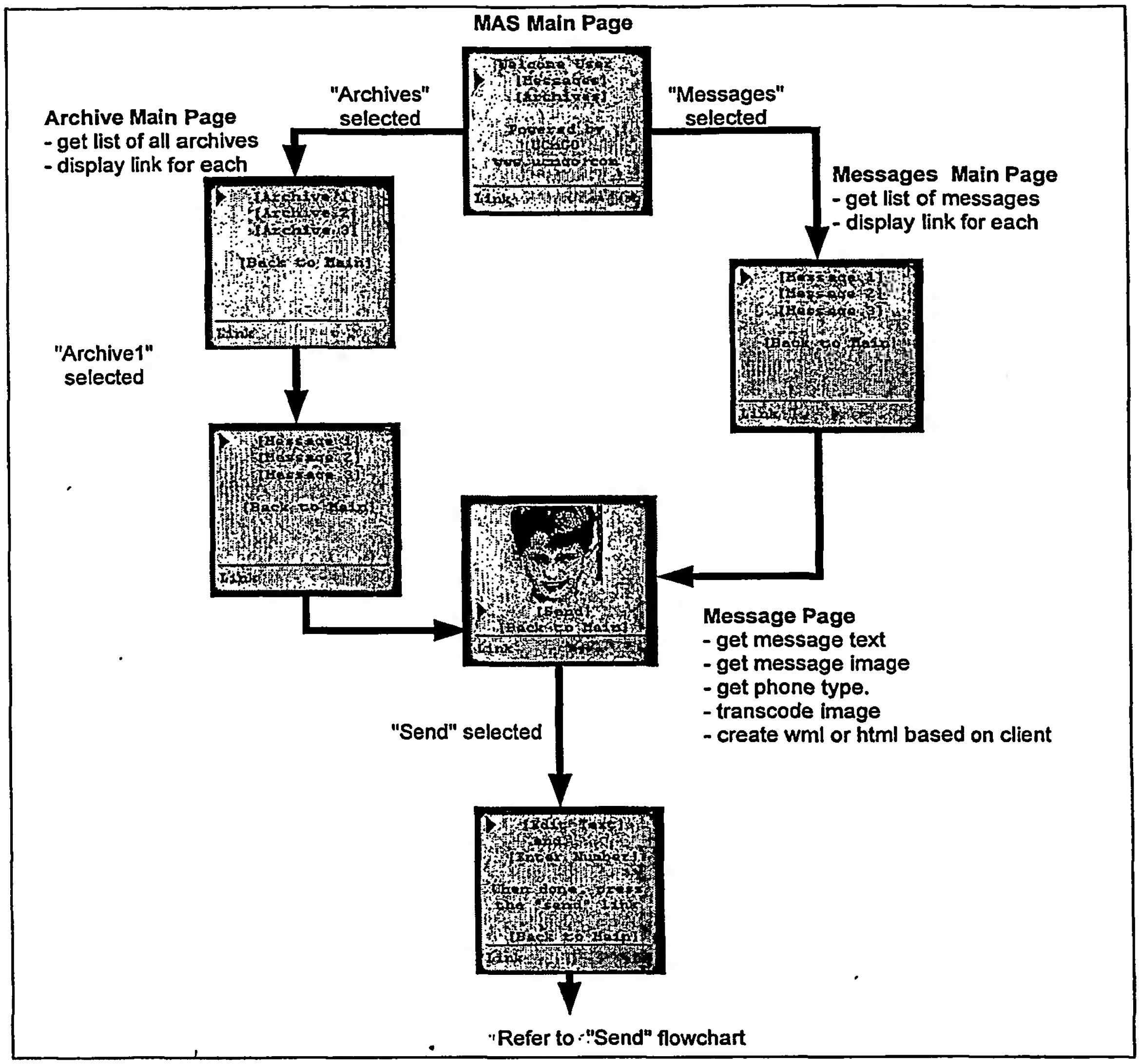

MAS Main Page
"Archives" selected
"Messages" selected
Archive Main Page
- get list of all archives
- display link for each
Messages Main Page
- get list of messages
- display link for each
"Archive1" selected
Message Page
- get message text
- get message image
- get phone type.
- transcode image
- create wml or html based on client
"Send" selected
Refer to "Send" flowchart

FIG. 13

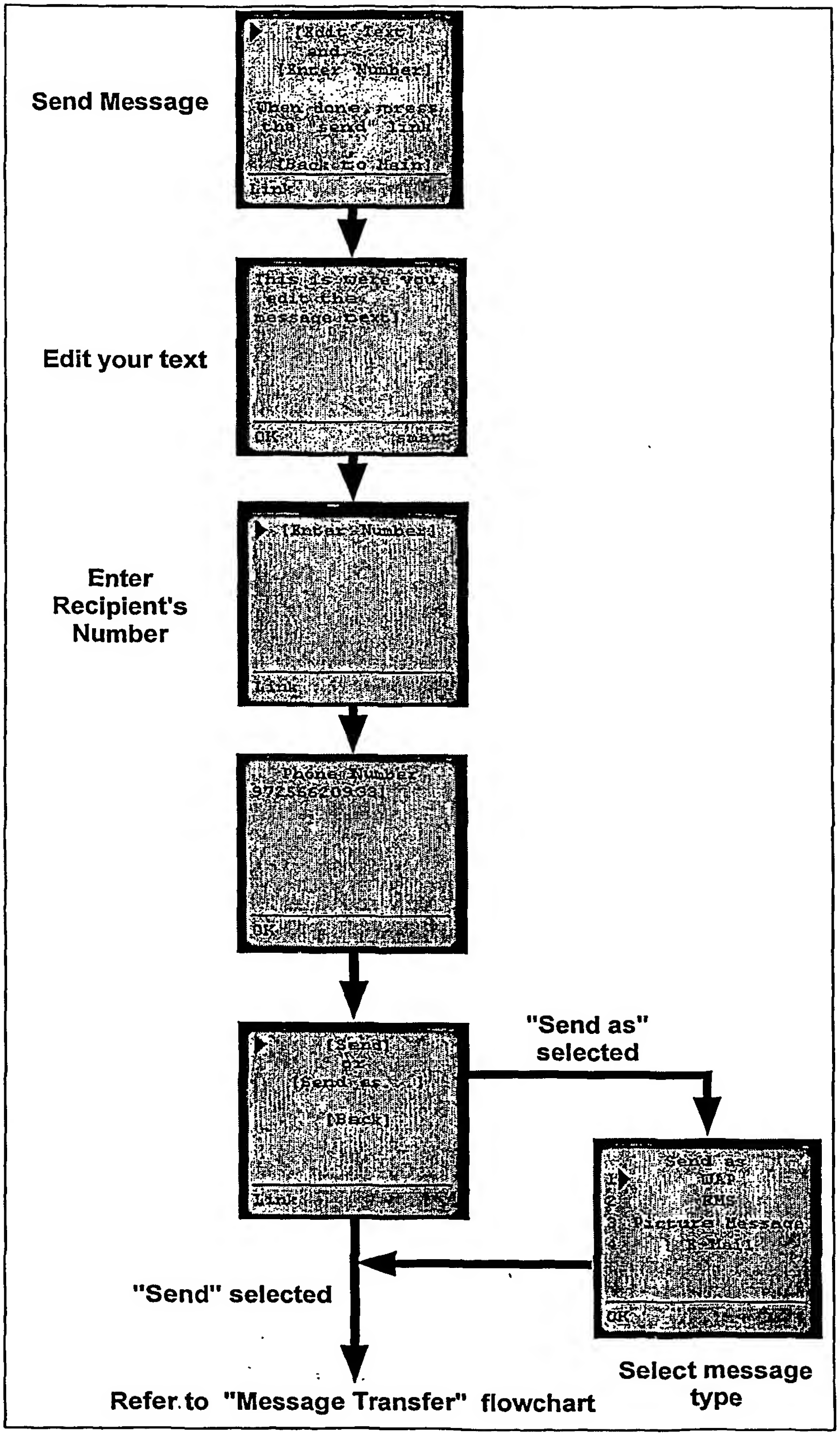

Send Message
[Add Text]
and
[Enter Number]
When done, press the "send" link
[Back to Main]
Link
Edit your text
This is were you edit the message text]
OK
Enter Recipient's Number
[Enter Number]
Link
Phone Number
972556209331
OK
[Send]
[Send as...]
[Back]
Link
"Send as" selected
Send as
WAP
SMS
Picture Message
OK
Select message type
"Send" selected
Refer to "Message Transfer" flowchart

FIG. 14

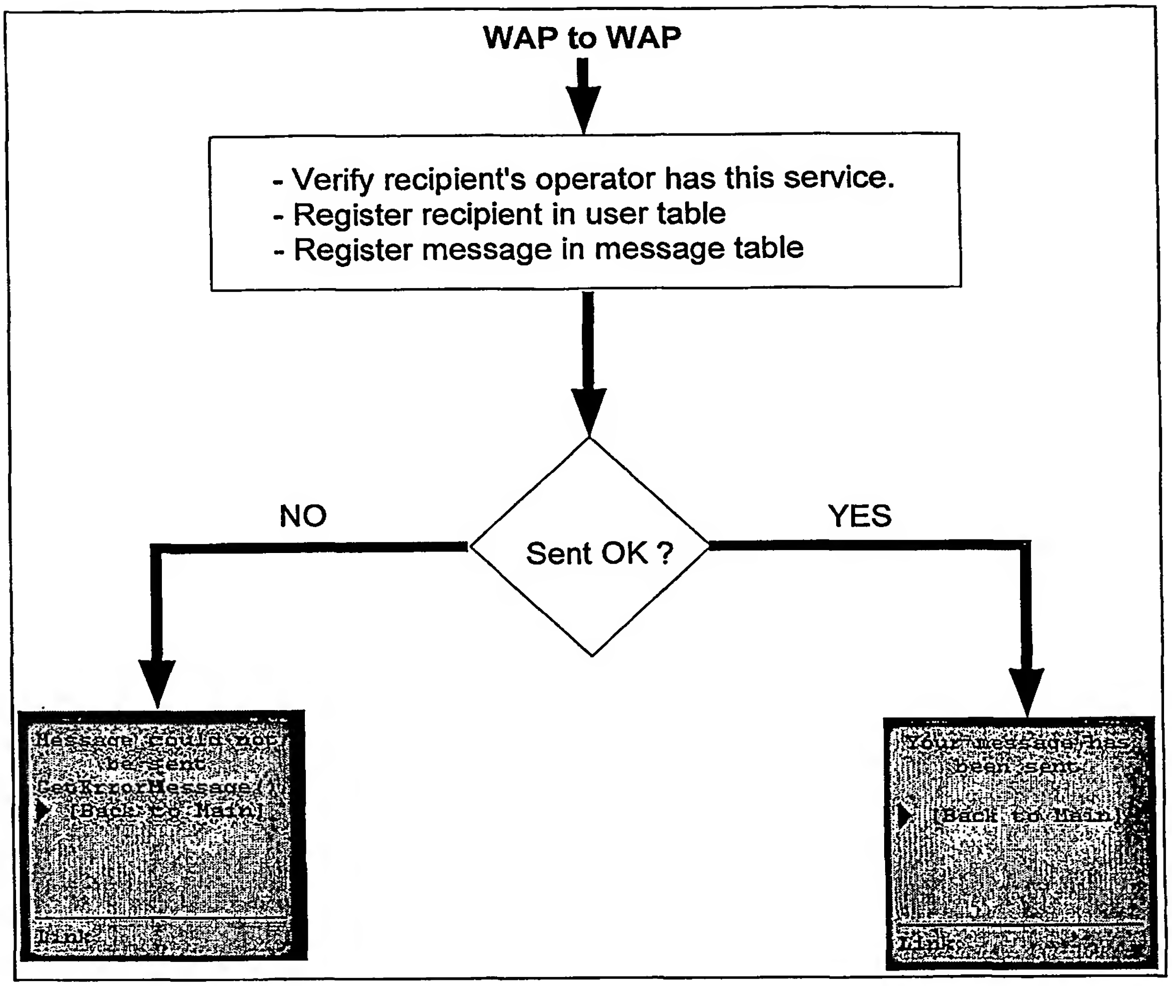

WAP to WAP
- Verify recipient's operator has this service.
- Register recipient in user table
- Register message in message table
Sent OK ?
NO
YES
Message could not be sent
[Back to Main]
Link
Your message has been sent
[Back to Main]
Link

FIG.15

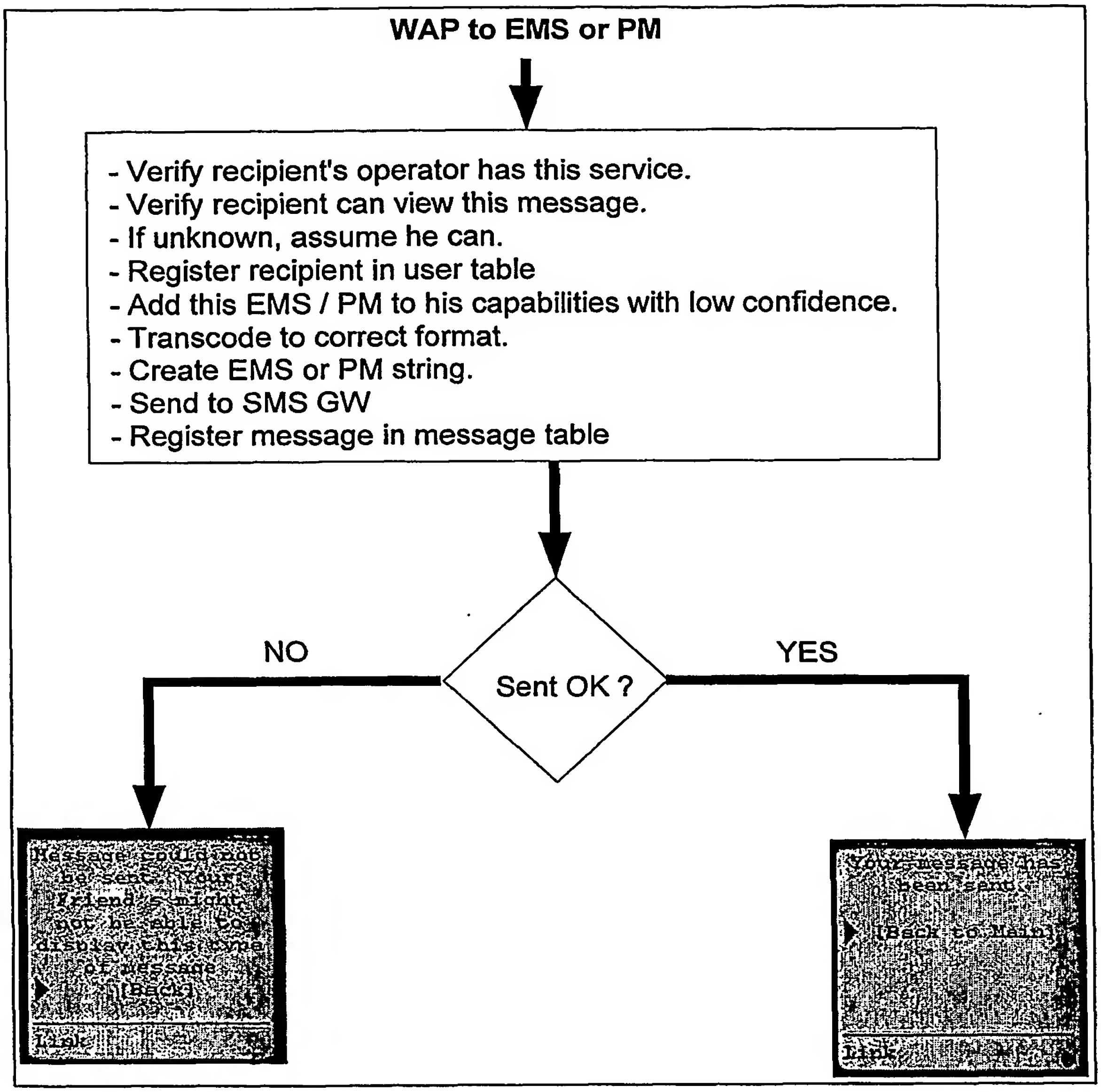

WAP to EMS or PM
- Verify recipient's operator has this service.
- Verify recipient can view this message.
- If unknown, assume he can.
- Register recipient in user table
- Add this EMS / PM to his capabilities with low confidence.
- Transcode to correct format.
- Create EMS or PM string.
- Send to SMS GW
- Register message in message table
Sent OK ?
NO
YES
Message could not be sent. Your friend's might not be able to display this type of message
[Back]
Link
Your message has been sent.
[Back to Main]
Link

Fig. 16

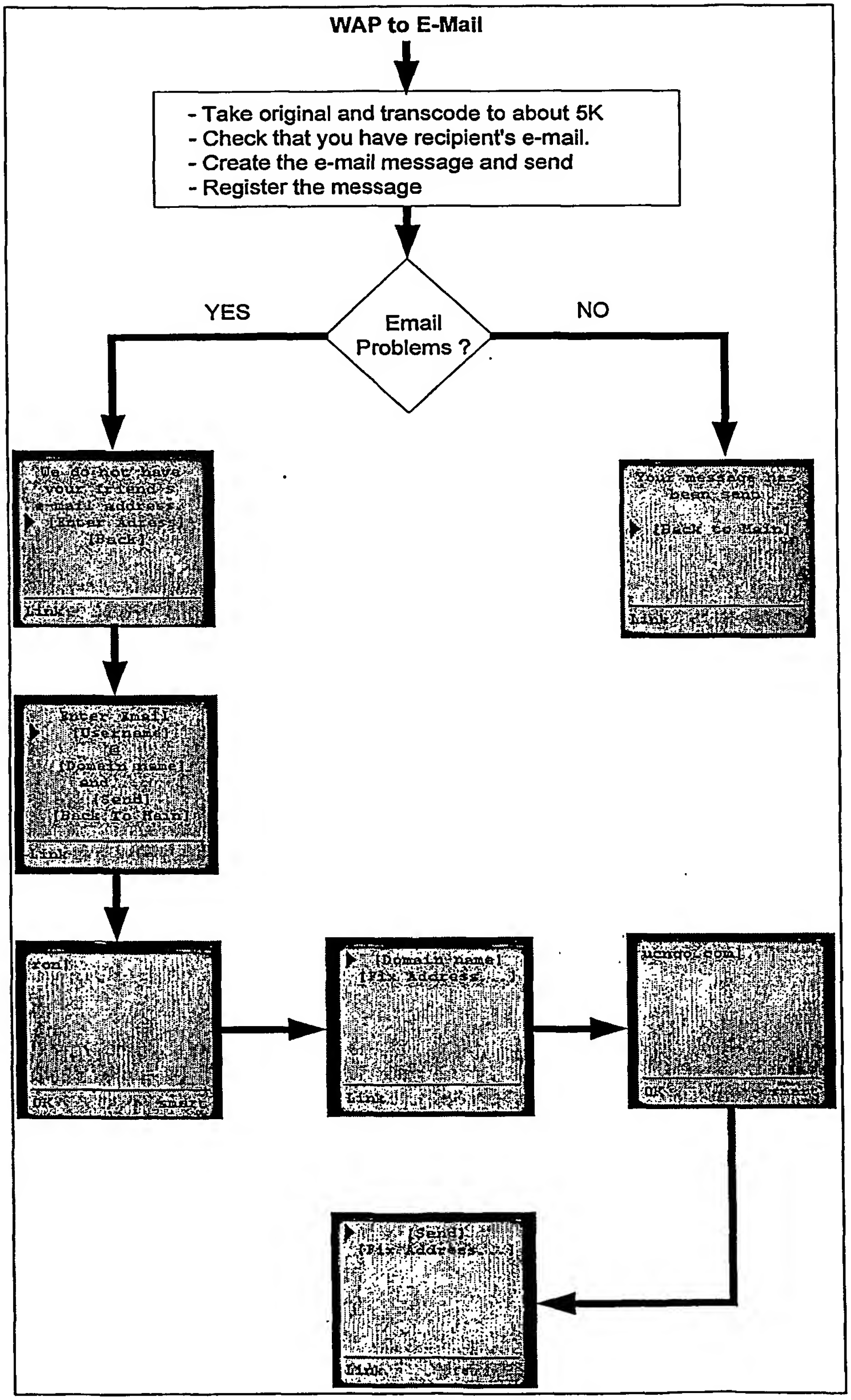

WAP to E-Mail
- Take original and transcode to about 5K
- Check that you have recipient's e-mail.
- Create the e-mail message and send
- Register the message
Email Problems ?
YES
NO
[Back]
[Back to Main]
[Username]
@
[Domain name]
[Send]
[Back To Main]
[Domain name]
[Send]

FIG. 17

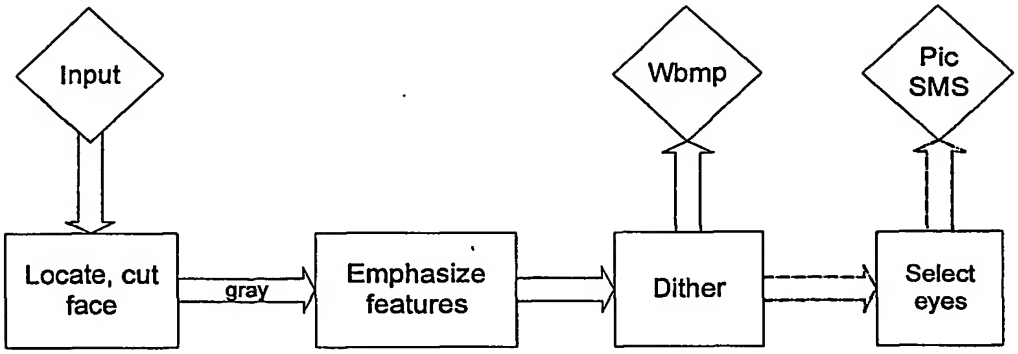

Input
Wbmp
Pic SMS
Locate, cut face
gray
Emphasize features
Dither
Select eyes

FIG.18

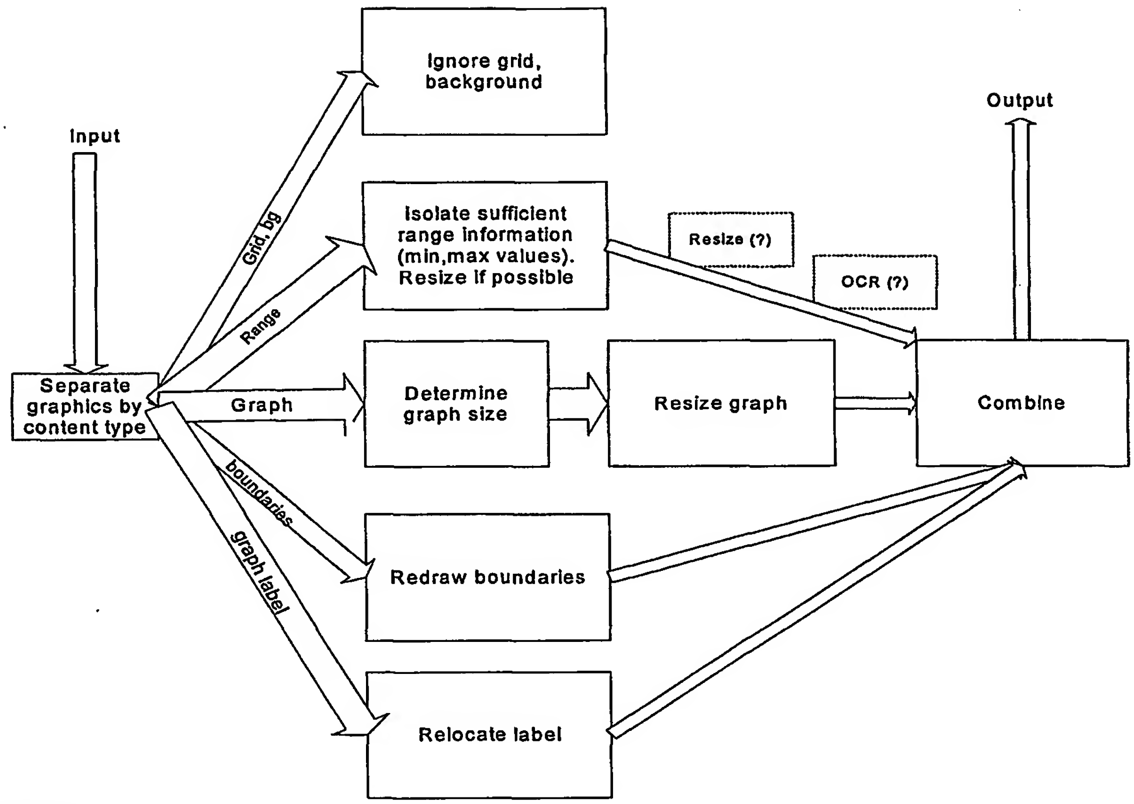

Ignore grid, background
Output
Input
Grid, bg
Isolate sufficient range information (min,max values). Resize if possible
Resize (?)
OCR (?)
Range
Separate graphics by content type
Graph
Determine graph size
Resize graph
Combine
boundaries
graph label
Redraw boundaries
Relocate label

FIG.19

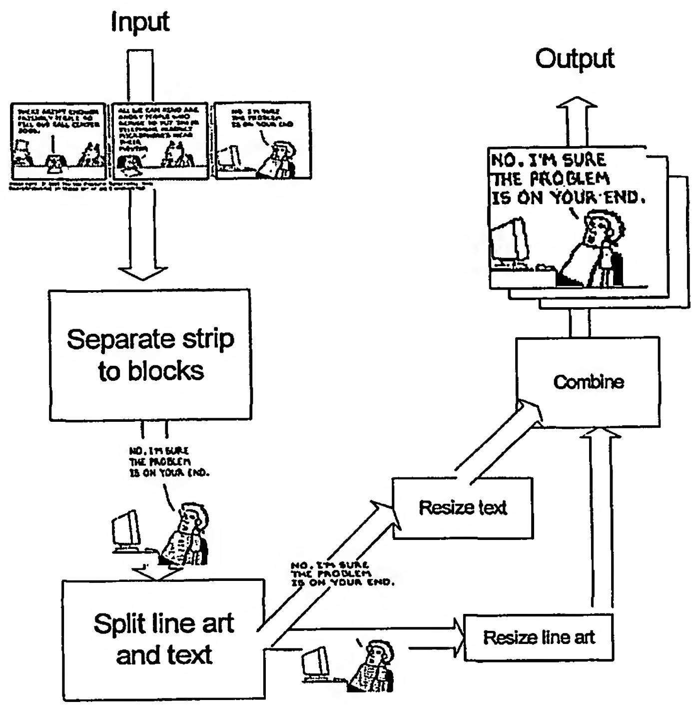

Input
Output
NO. I'M SURE
THE PROBLEM
IS ON YOUR END.
Separate strip
to blocks
Combine
Resize text
Split line art
and text
Resize line art

FIG.20

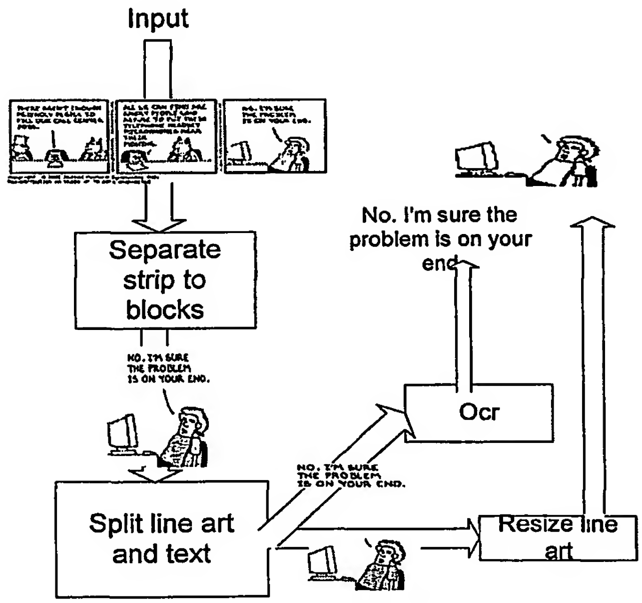
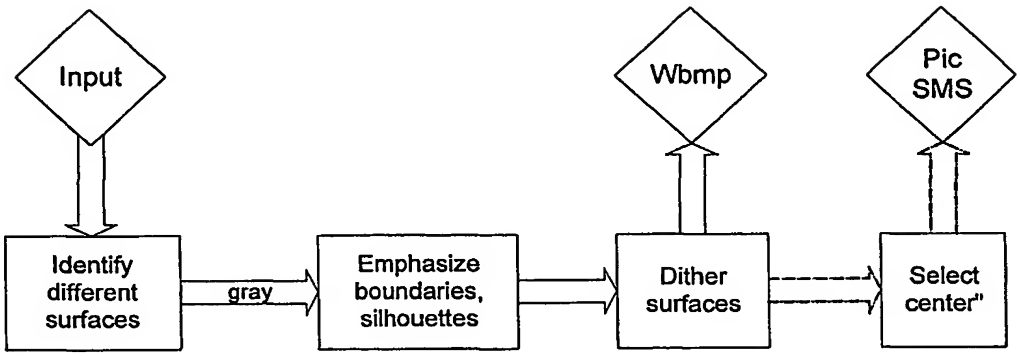

Input
Separate strip to blocks
NO. I'M SURE THE PROBLEM IS ON YOUR END.
Split line art and text
NO. I'M SURE THE PROBLEM IS ON YOUR END.
Ocr
No. I'm sure the problem is on your end
Resize line art

FIG.21

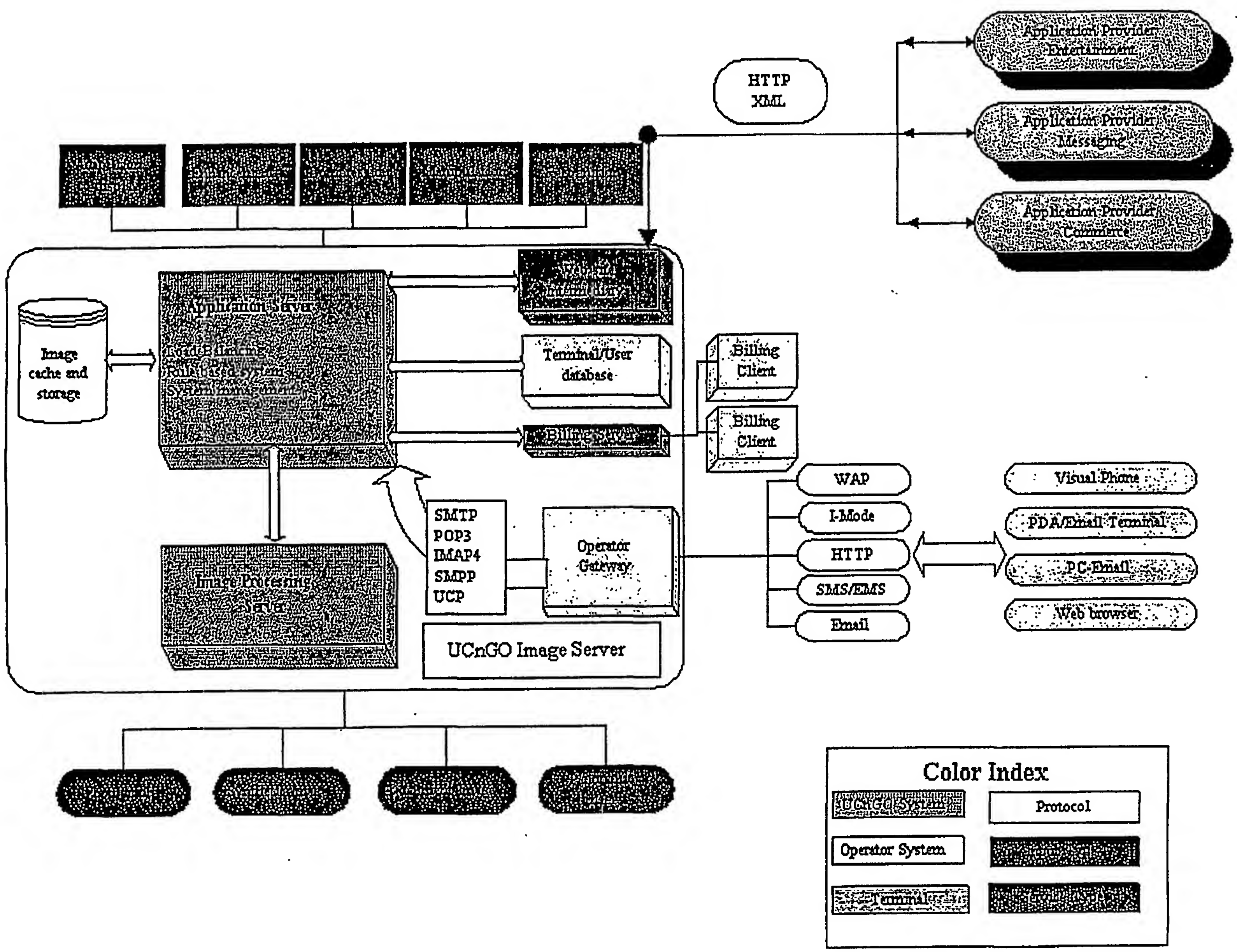


Input
Wbmp
Pic
SMS
Identify
different
surfaces
gray
Emphasize
boundaries,
silhouettes
Dither
surfaces
Select
center"

FIG.22

FIG. 23

System Architecture



```
<smil>
    <head>
        <meta name="title" content="mms" />
        <meta name="author" content="John Smith" />

        <layout>  <! --This an "landscape" screen (2*qcif)-->
                <root-layout width="352" height="144"/>
                <region width="176" height="144" left="0" top="0" />
                <region width="176" height="144" left="176" top ="0"/>
        </layout>

<!--   <layout>   // This is a "portrait" screen -->
<!--            <root-layout width="176" height="216"/>  -->
<!--            <region id="Image" width="176" height="144" left="0" top="0" />  -->
<!--            <region id="Text" width="176" height="72" left="0" top ="144"/>  -->
<!--   </layout>  -->

    </head>
    <body>
      <par dur = "8s">
        <img  src = "FirstImage.jpg" region="Image" />
        <text src = "FirstText.txt" region="Text" />
        <audio src = "FirstSound.amr"/>
      </par>
      <par dur = "7s" >
        <img   src = "SecondImage.jpg" region="Image" />
        <text  src = "SecondText.txt" region="Text" />
        <audio src = "SecondSound.amr"/>
      </par>
      <par dur = "4s" >
        <img   src = "ThirdImage.jpg" region="Image"/>
        <text  src =  "ThirdText.txt" region="Text"/>
        <audio src = "ThirdSound.amr"/>
      </par>
    </body>
</smil>
```

FIG. 24

4.3 Presentation.

SMIL is a *presentation* format, i.e. a SMIL page contains information about the appearance of different multimedia elements on a display. When SMIL is used to represent content on a PC screen, normally a window is opened whose size is defined by the layout element of the SMIL page to be displayed. In this way, the appearance of the SMIL page on the screen will reflect exactly the organization of the content as the author had created it.

When SMIL is used for the presentation of multimedia messages on mobile terminals, the size of the window is severely limited by the resolution and appearance of the terminal display. The *layout* of a multimedia message represents the content as created by the originator, but it is well possible that the original layout simply does not fit into the display of the receiving terminal. Therefore, SMIL exchange must be simple enough to ensure that -if the displays of the originator and receiver terminal are different- the content can still be displayed, possibly by changing the relative position of the different elements.

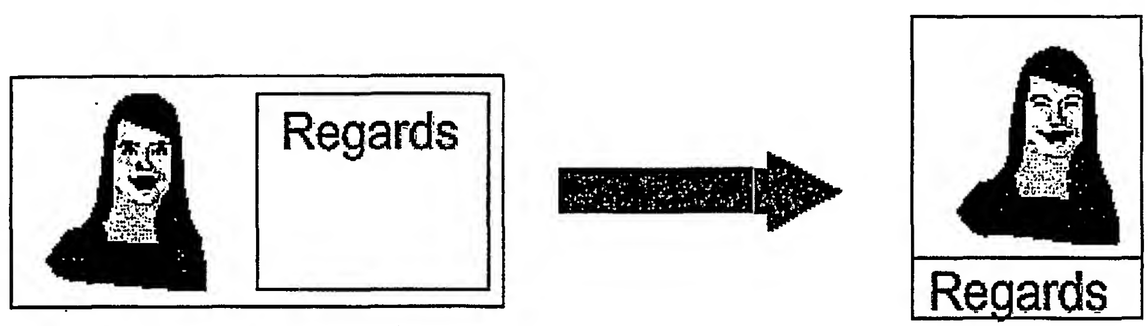


Fig. 25

Face detection for Subimage Selection

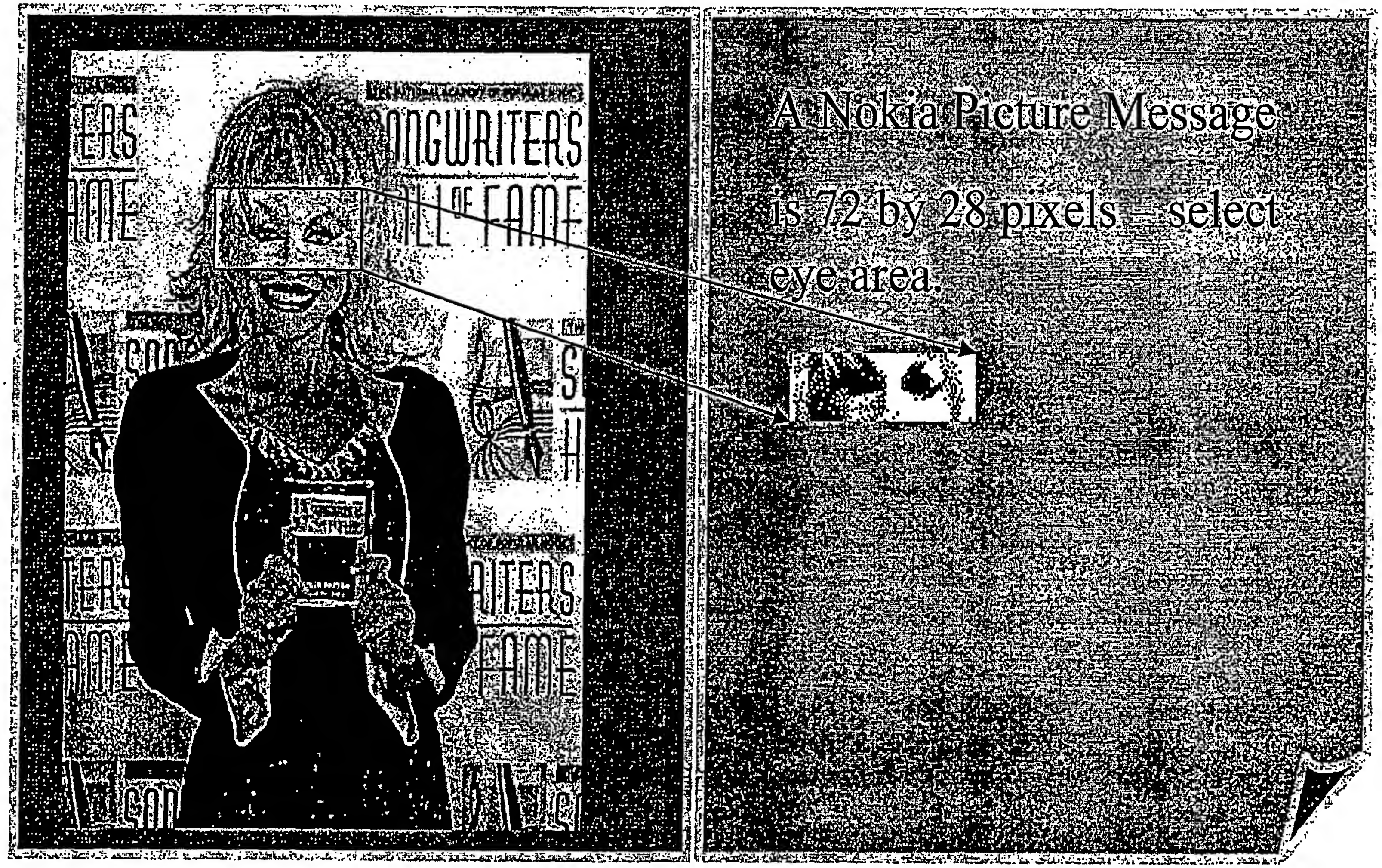


FIG. 26

Face Detection & Image Adaptation Features

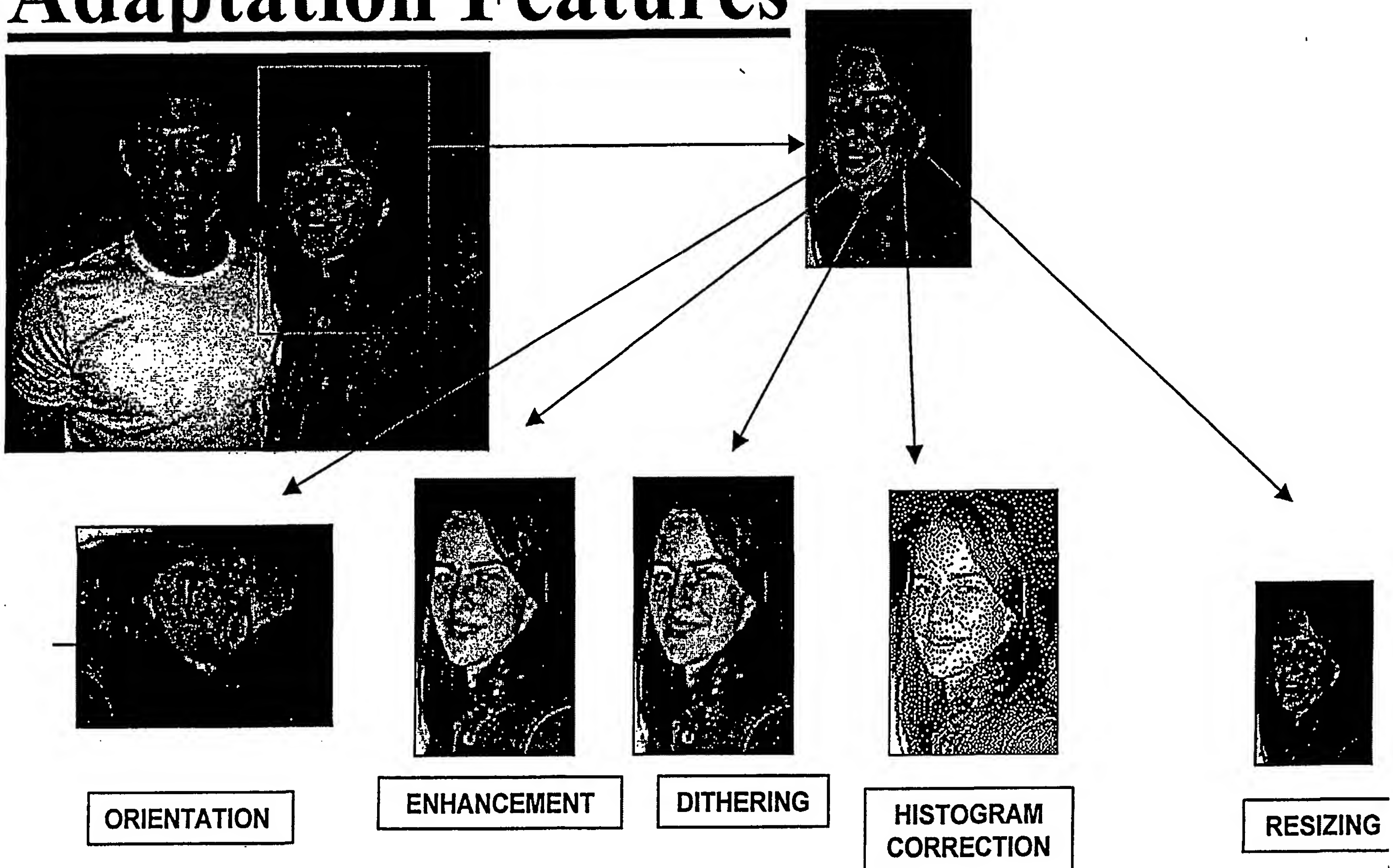


FIG. 27

Face Detection

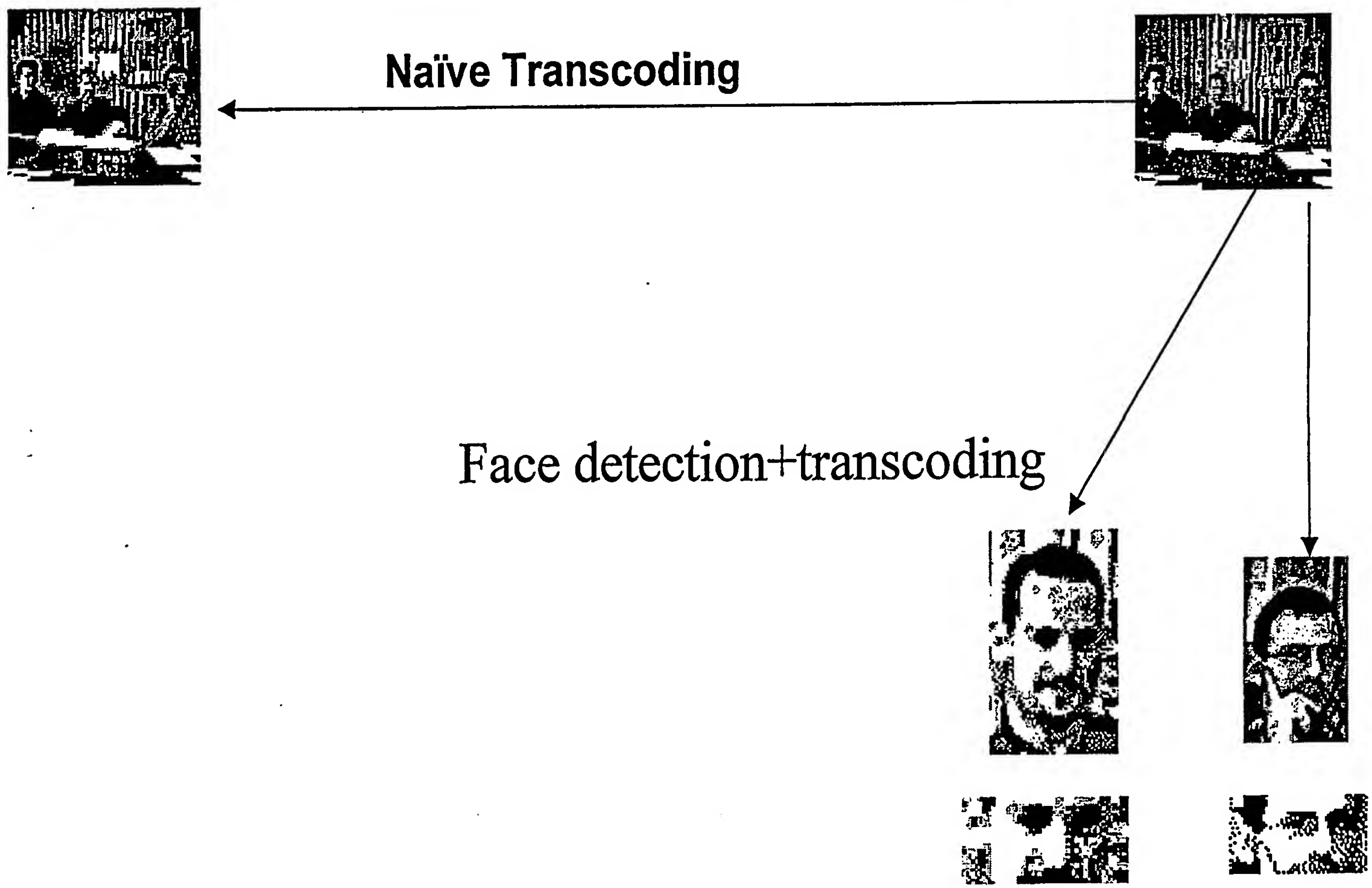
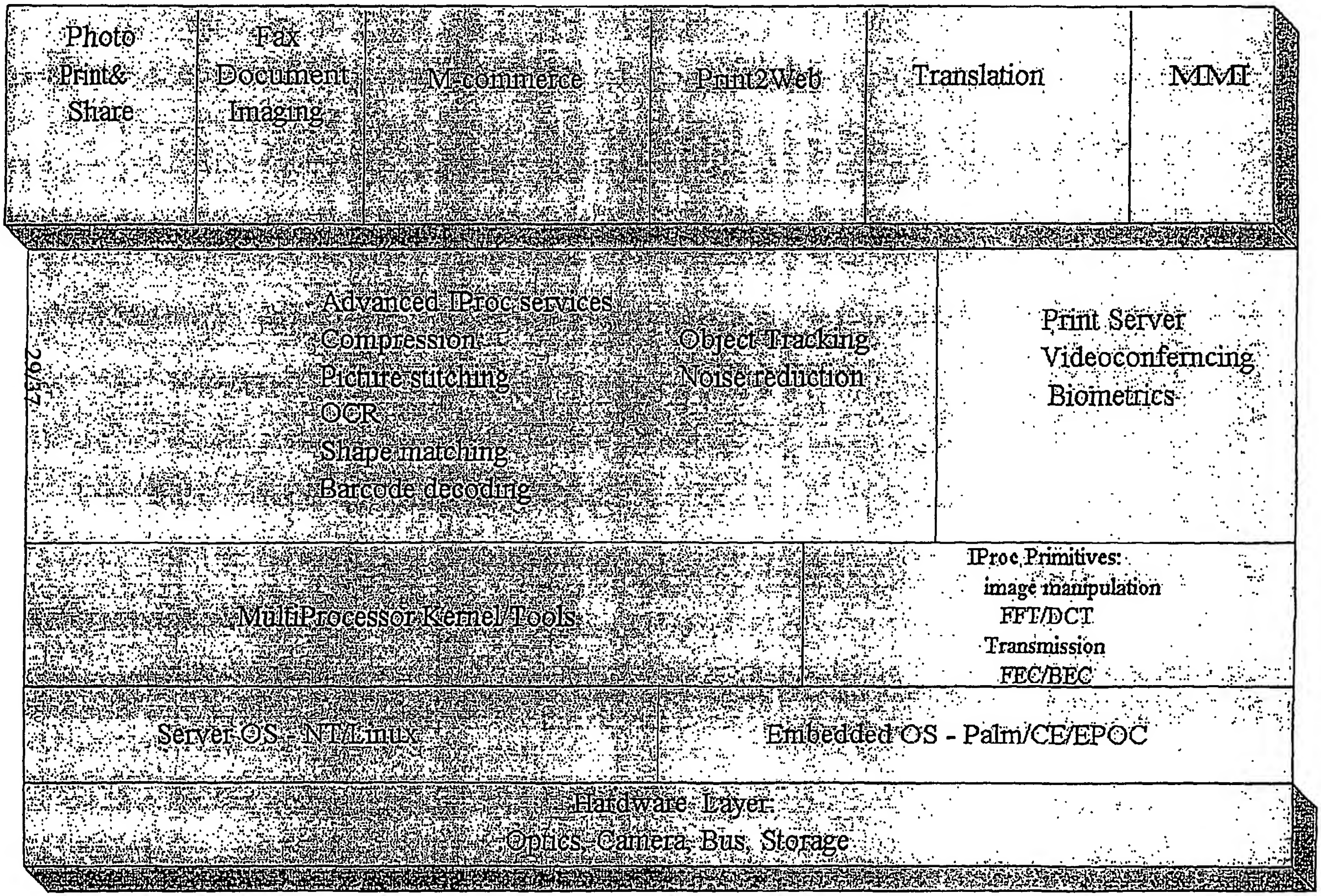


FIG 28

UCnGO Technology Platform



Presentation I · 13

FIG. 29

Fig. 30

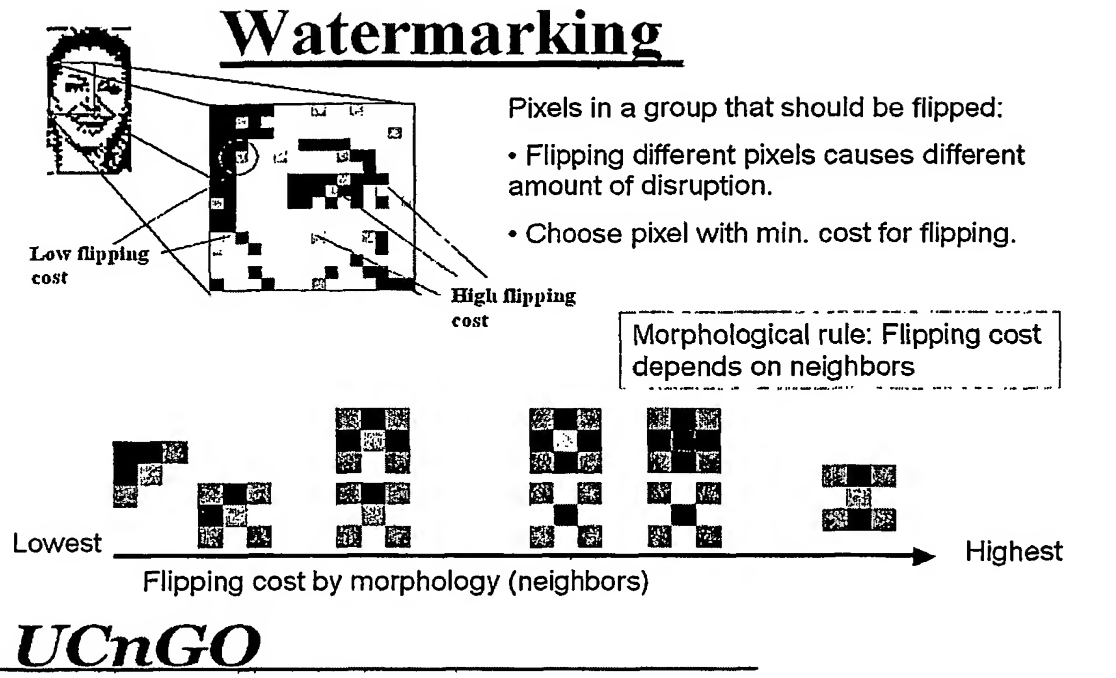


Figure XX: Watermarking algorithm for B/W images (picture messages, WBMP)

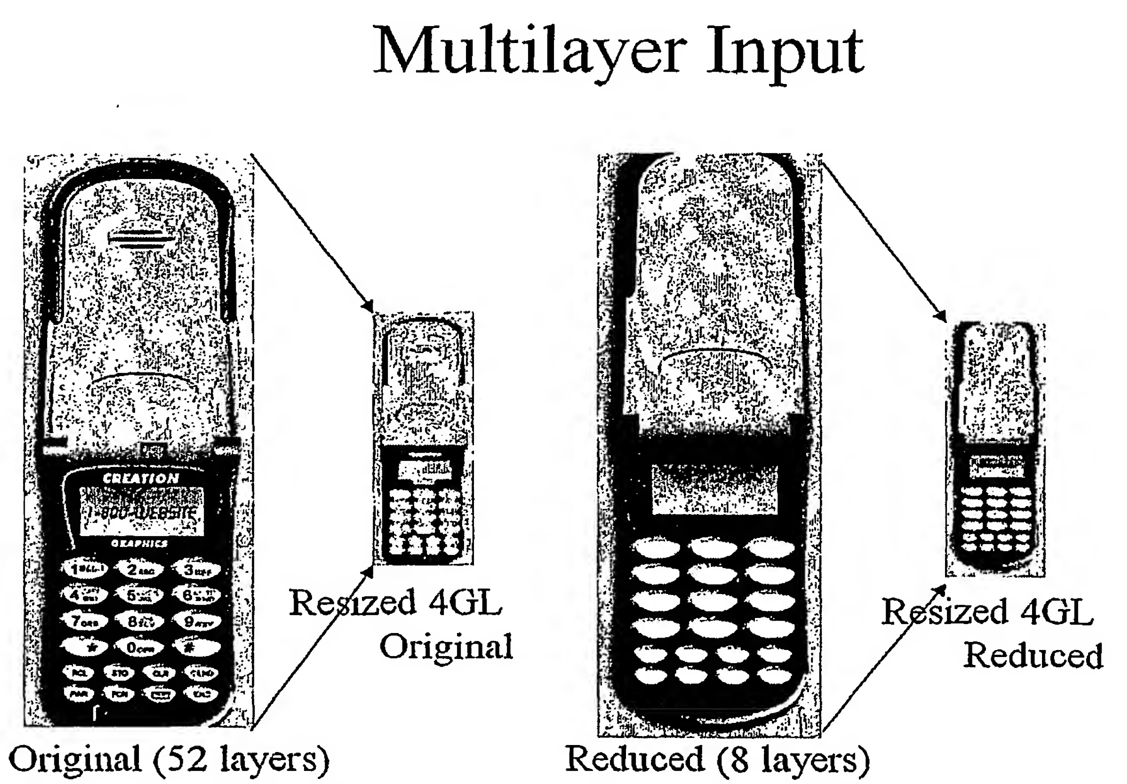


Figure XX: Layer reduction and its effect on the message appearance

Fig. 31

Content Based Conversion

Stock Charts:

Remove superfluous information

Maintain line solidity

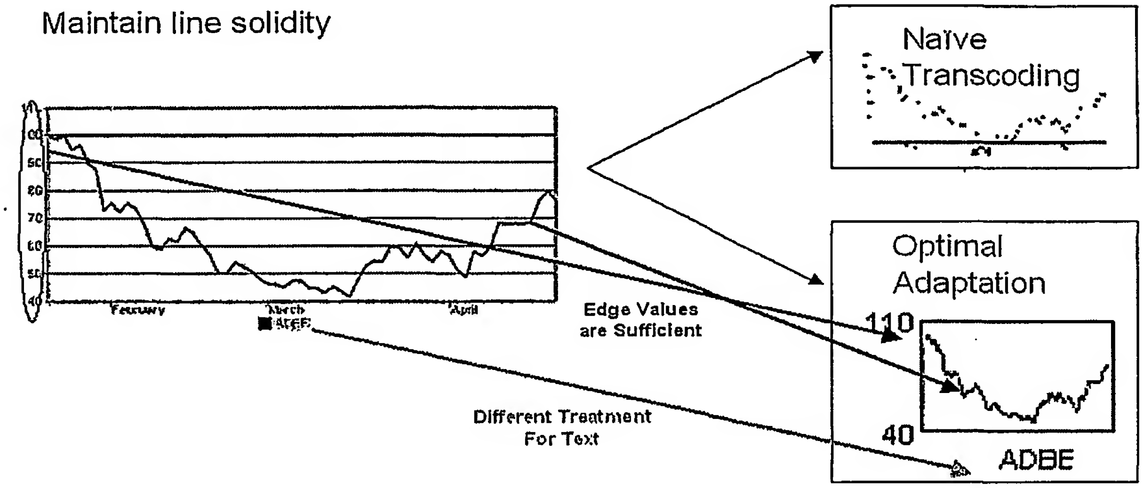


Figure XX: content based conversion for charts (color Line-art)

Content Based Conversion

Cartoons:

Different processing for text and drawings

Ignore gray level information

Maintain line solidity

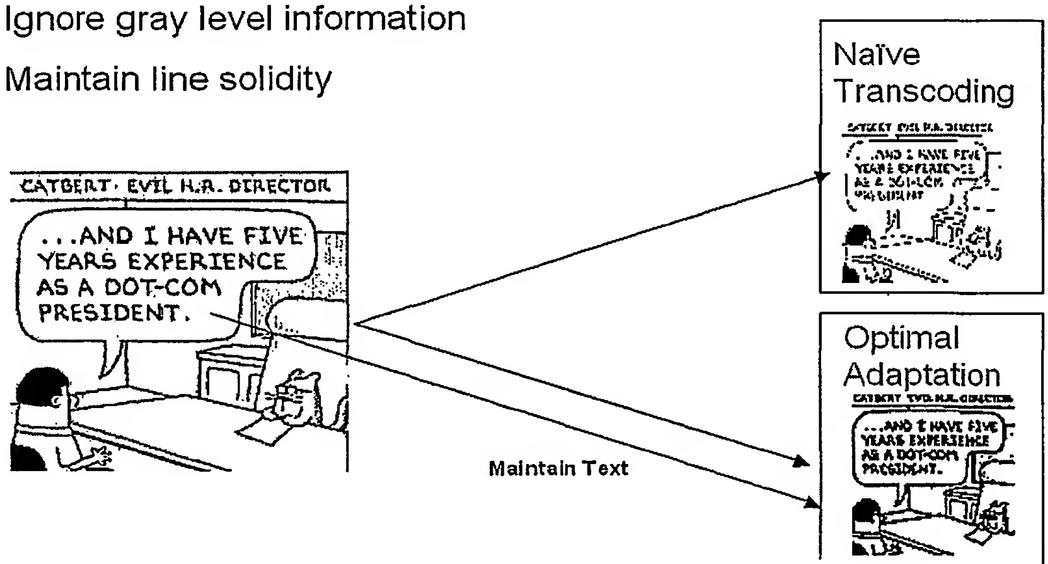


Figure XX: content based conversion for cartoons (B&W Line-art)

Fig. 32

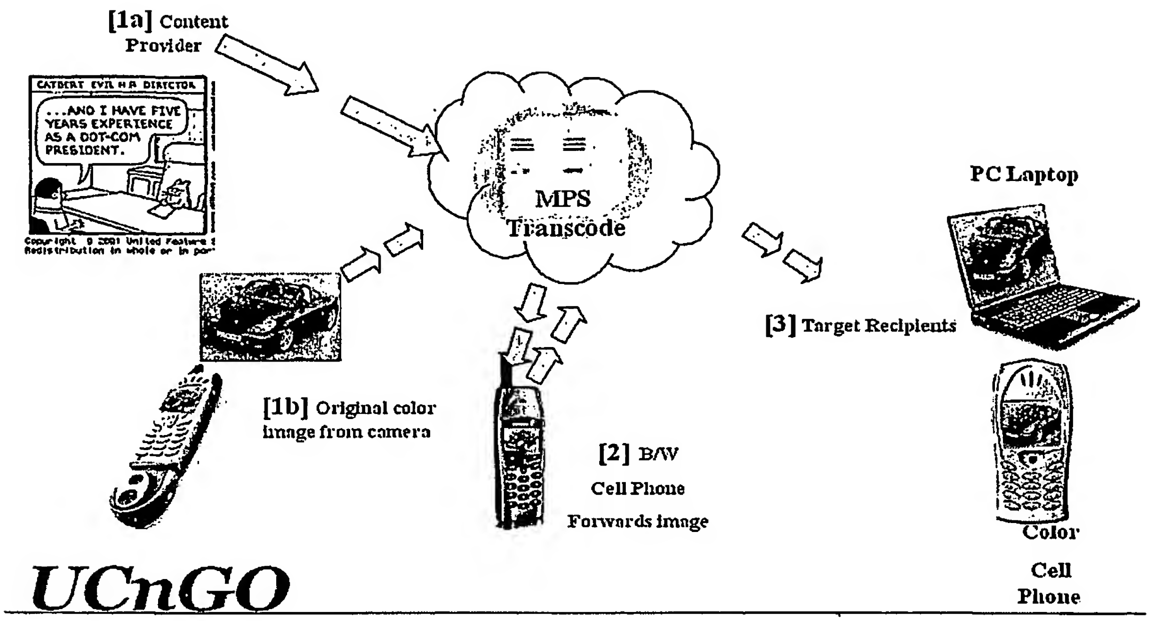


Figure: cases where caching and smart-forwarding are required.

Fig. 33

Specifications

Implementation: The caching and smart forwarding system is based on a web-caching mechanism – converted and original media objects are stored in a specially configured set of web-caches, and are retrieved by URI. The basic operation scenario is depicted in the figures below.

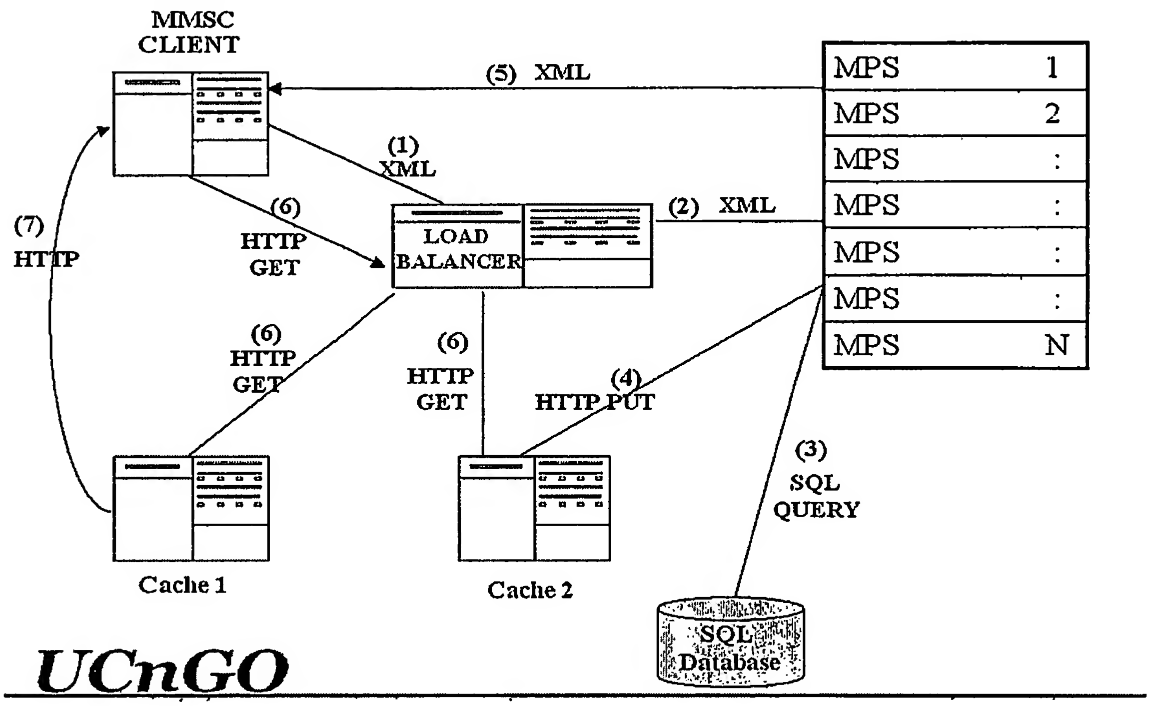


Figure: Sequence of operations for a cached transaction

Algorithmic Cache Sequence

- For incoming object A (after SMF) + filtered Target Parameters calculate MD5 "C" plus target MIME type "D".
- Perform HTTP HEAD request to check if in cache. request filename is "C"."D".
- If not in cache, perform transcode, perform HTTP PUT for transcode result in cache.
- Return URI.

Figure: sequence of operations for caching

Fig. 34

Algorithmic SMF Sequence

- For incoming object A calculate MD5, extract internal hash "C", MIME type "D" from SQL database (indexed query).
- If new object, put in database as original content, perform HTTP PUT for original object in cache.
- If existing object, perform HTTP HEAD request to check if in cache. Object filename is "C"."D". Update last requested counter.
- If in cache, return URI.
- If not in cache, put the new input object A as file with the original content hash value "C" as the file name.

Figure: sequence of operations for smart-forwarding

Hash Database

- Any SQL database can be used, e.g. MySQL.
- Single indexed table (index on key value)/ double indexed table (last requested update).
- Periodic cleanup required of least accessed objects.
- Table can reside in RAM – e.g. for 10 Million cached messages, less than 4Gb of RAM required.
- Can be implemented as an indexed linked list with no database if database performance issues arise.
- Database scaling is simple based on hash value partitioning (e.g. server number 1 handles hash values less than TBD etc.).

Figure: Hash database connecting transcoded objects and original objects

Fig. 35

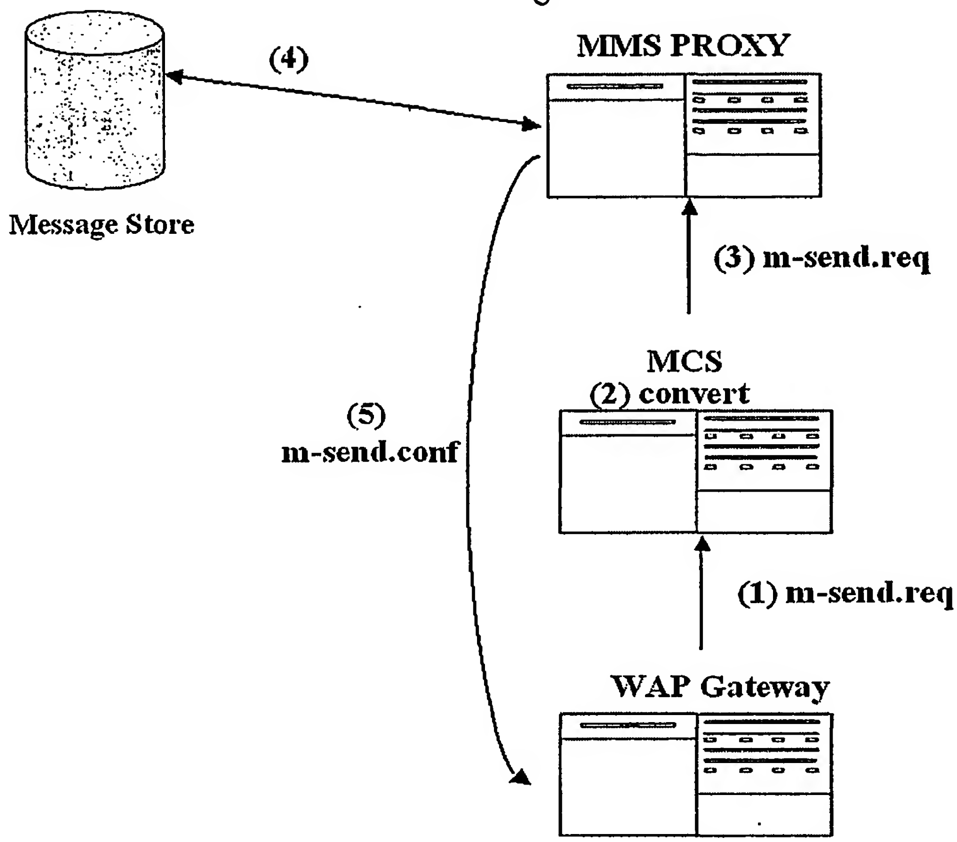


Figure XX: MM1 message interface to the UCnGO MCS

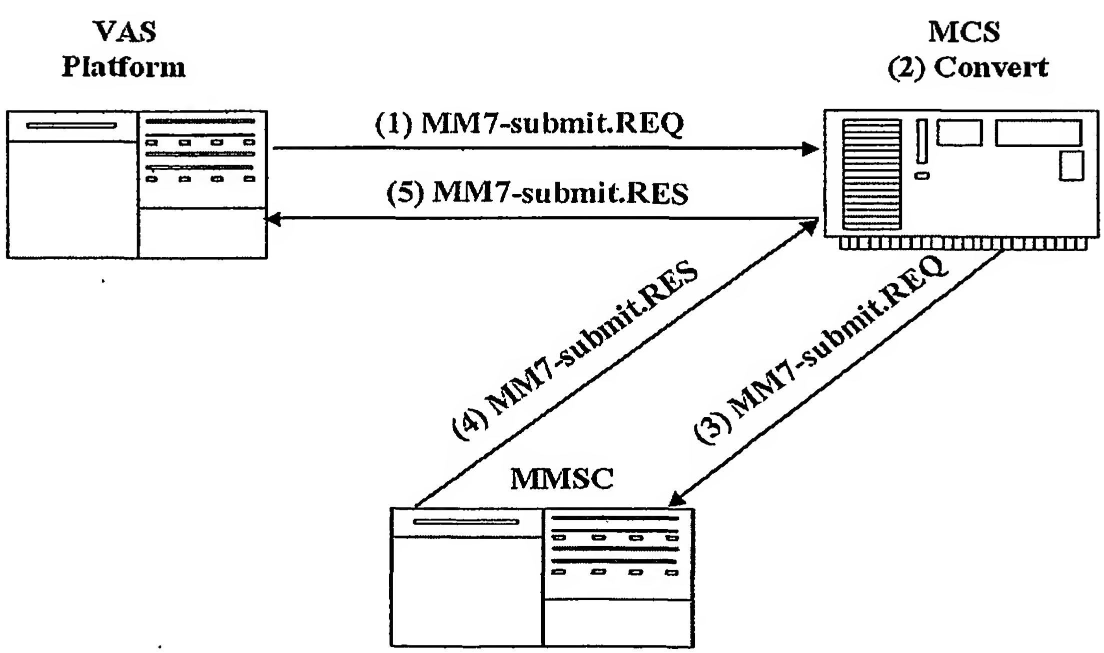


Figure XX: MM7 message interface to the UCnGO MCS

Fig. 3

```
S:    <wait for connection on TCP port 25>
C:    <open connection to server>
S:    220 mmsc.ucngomcs.net Simple Mail Transfer Service Ready
C:    EHLO cnn.com
S:    250 mmse.ucngomcs.net
S:    250 SIZE 150000
C:    MAIL FROM: <cnn.com> SIZE=50000
S:    250 OK
C:    RCPT TO: <+3508401234567@mmsc.operator.net>
S:    250 OK
C:    DATA
S:    354 Start mail input ; end with <CR><LF>.< CR><LF>
C:    Date: 01 November 2001 00:00 EST
C:    from: cnn . com
C:    to: +3508401234567@mmsc.operator.net
C:    subject: Newscast with text and animated gif
C:    Content-type: multipart/mixed
C:    X-Mms-Message-Class: personal
C:    X-Mms-Expiry: 05 November 2002 10:00 EST
C:    X-Mms-delivery-time: 05 November 2002 09:00 EST
C:    X-Mms-Priority: Normal
C:    ——=_Part_0_9840522.997734797363
C:    Content-Type: text/plain; charset=us-ascii
C:    Unisys launches MMS video services in Portugal ...
C:    ——=_Part_0_9840522.997734797363
C:    Content-Type: image/gif;name="girl_video.gif"
C:    Content-Disposition: attachment; filename=girl_video.gif
C:    ——=_Part_0_9840522.997734797363
C:    Content-Type: image/GIF
C:    Content-Transfer-Encoding: base64
C:    jICYMTAyMQCNkIkagDE0Ny4xMTcuMjA5Ljk3L1RZUEU9SVB2NACPgJ E3Lj
...
...
C:    bmcgdGHLICJQcmlvcml0eSIgZmLLbGQuIEL0IGLzIHB1dCBhcyAiTG93Ii4NC
C:    g==
C:    ——=_Part_0_9840522.997734797363
C:    <CR><LF>.< CR><LF>
S:    250 OK
C:    QUIT
S:    221 mmsc.ucngomcs.net Service closing transmission channel
```

Figure: Sample MMS submission for transcoding from a VAS through MM7

Fig. 37

UCnGO Media Transcoding Matrix